March 2012 ניסן התשע"ב – מרס 2012

ICS CODE: 35.020

טכנולוגיית המידע – הגדרות מונחים:
תכנות מחשבים

Information technology – Vocabulary: Computer Programming

מכון התקנים הישראלי
The Standards Institution of Israel

רח' חיים לבנון 42, תל-אביב 69977, טל' 03-6465154, פקס' 03-6412762, www.sii.org.il דק 20/6/12

תקן זה הוכן ואושר על ידי הוועדה הטכנית 2110 – מונחי טכנולוגיית המידע, בהרכב זה:

אוניברסיטת בר-אילן	-	אריאל פרנק
אוניברסיטת תל-אביב	-	אורה קפלן
איגוד האינטרנט הישראלי	-	יונתן בן אברהם, דורון שקמוני
איגוד ישראלי לעיבוד אינפורמציה (איל"א)	-	מנחם דישון
האקדמיה ללשון העברית	-	מלכה זמלי, מלכה מוצ'ניק, אורה שורצולד
יבמ ישראל בע"מ	-	שלמה מסטבאום
לשכת מנתחי מערכות	-	אמנון שפירא (יו"ר)
מיקרוסופט ישראל	-	תמר שגיא
רשות השידור	-	דוד גומבוש
רשות ההסתדרות לצרכנות	-	יובל פרידמן

כמו כן תרמו להכנת התקן: אבישי בידני, דורון פינקלשטיין, חיים הירשהורן, יבגני זינגרמן, מיכל שחר, שמשון הופמן, שרה חשבי.

דנית לביא ריכזה את עבודת הכנת התקן.

הודעה על רוויזיה

תקן זה בא במקום

התקן הישראלי ת"י 1080 חלק 7 מדצמבר 1998

גיליון התיקון מס' 1 מיולי 2006

מילות מפתח:

עיבוד נתונים, עיבוד מידע, אוצר מילים, מונחים, הגדרות, טכנולוגיית המידע

Descriptors:

data processing, information processing, vocabulary, terms, definitions, information technology

עדכניות התקן

התקנים הישראליים עומדים לבדיקה מזמן לזמן, ולפחות אחת לחמש שנים, כדי להתאימם להתפתחות המדע והטכנולוגיה.

המשתמשים בתקנים יוודאו שבידיהם המהדורה המעודכנת של התקן על גיליונות התיקון שלו.

מסמך המתפרסם ברשומות כגיליון תיקון, יכול להיות גיליון תיקון נפרד או תיקון המשולב בתקן.

רשמיות התקן

יש לבדוק אם המסמך רשמי או אם חלקים ממנו רשמיים. תקן רשמי או גיליון תיקון רשמי (במלואם או בחלקם) נכנסים לתוקף 60 יום מפרסום ההודעה ברשומות, אלא אם בהודעה נקבע מועד מאוחר יותר לכניסה לתוקף.

סימון בתו תקן

כל המייצר מוצר, המתאים לדרישות התקנים הישראליים החלים עליו, רשאי, לפי היתר ממכון התקנים הישראלי, לסמנו בתו תקן:

זכויות יוצרים

© שעתוק של מונחים והגדרות הנכללים בתקן זה מותר במדריכים לימודיים, בחוברות הוראות, בפרסומים טכניים ובכתבי עת - למטרות חינוכיות או יישומיות בלבד.

התנאים לשעתוק כזה הם:

לא יוכנס כל שינוי במונחים ובהגדרות;

שעתוק כזה אינו מותר עבור מילונים או פרסומים דומים המוצעים למכירה;

תקן זה יאוזכר כמסמך המקור.

למעט האמור לעיל אין לשעתק או להשתמש בחלק כלשהו של פרסום זה, בכל צורה שהיא או בכל אמצעי שהוא, אלקטרוני או מכני, לרבות צילום וזיעור, ללא רשות בכתב מאת מכון התקנים הישראלי.

תוכן העניינים

הקדמה 1

07.01 - סוּגֵי שָׂפוֹת - Kinds of languages 3

07.02 - שִׁיטוֹת, טֶכְנִיקוֹת וּמִבְנֶה - Methods, techniques and structure 9

07.03 - חִזָּרוּר וּנְסִיגָה - Iteration and recursion 12

07.04 - הֲכָנַת תָּכְנִיּוֹת - Program preparation 14

07.05 - קִשּׁוּר וּטְעִינָה - Linking and loading 21

07.06 - בִּצּוּעַ תָּכְנִית - Program execution 24

07.07 - נִפּוּי וּבִדּוּק - Debugging and checking 30

07.08 - תִּכְנוּת מִיקְרוֹ - Microprogramming 34

07.09 - פְּקֻדּוֹת וּמְעָנִים - Instructions and addresses 35

07.10 - תַּהֲלִיכִים בּוֹ-זְמַנִּיִּים - Concurrent processes 41

07.11 - סְבִיבוֹת תְּמִיכָה - Support environments 43

07.12 - מַטָּרוֹת וְעֶקְרוֹנוֹת - Goals and principles 45

איורים 48

מפתח מונחים א-ב 52

מפתח מונחים A-B 64

הקדמה

תקן זה הוא חלק מסדרת התקנים הישראליים ת"י 1080, המביאה מונחים והגדרות מהתחומים השונים של טכנולוגיית המידע. סדרת התקנים ת"י 1080 מבוססת על סדרת התקנים הבין-לאומיים ISO 2382 המתעדכנת מפעם לפעם, ועל מונחים חדשים הנאספים על ידי הוועדה ממקורות שונים. המונחים הנשאבים מהמקורות השונים משמשים להכנת חלקים חדשים או לעדכון חלקים קיימים.

המונחים מובאים באנגלית ובעברית, וההגדרות - בעברית.

חלקי סדרת התקנים ת"י 1080 שיצאו עד כה לאור הם אלה:

ת"י 1080 חלק 1 - מונחים בסיסיים
ת"י 1080 חלק 2 - פעולות אריתמטיות ולוגיות
ת"י 1080 חלק 3 - טכנולוגיית חומרה
ת"י 1080 חלק 4 - ארגון נתונים
ת"י 1080 חלק 5 - ייצוג נתונים
ת"י 1080 חלק 6 - הכנת נתונים ועיבודם
ת"י 1080 חלק 7 - תכנות מחשבים
ת"י 1080 חלק 8 - בקרה, כלילות ואבטחה
ת"י 1080 חלק 9 - תקשורת נתונים
ת"י 1080 חלק 10 - טכניקות ואַפשוֹרות של פעולה
ת"י 1080 חלק 11 - בקרה, קלט-פלט וציוד אריתמטי
ת"י 1080 חלק 12 - ציוד היקפי
ת"י 1080 חלק 13 - גרפיקה מחשבית
ת"י 1080 חלק 14 - אמינות, תחזוקתיות וזמינות
ת"י 1080 חלק 15 - שפות תכנות
ת"י 1080 חלק 16 - תורת המידע
ת"י 1080 חלק 17 - מסדי-נתונים
ת"י 1080 חלק 18 - עיבוד נתונים מבוזר
ת"י 1080 חלק 19 - תקשורת נתונים
ת"י 1080 חלק 20 - פיתוח מערכות
ת"י 1080 חלק 23 - עיבוד תמלילים
ת"י 1080 חלק 24 - ייצור כליל-מחשב
ת"י 1080 חלק 25 - רשתות מקומיות (LAN)
ת"י 1080 חלק 26 - חברור מערכות פתוחות (OSI)
ת"י 1080 חלק 27 - המשרד הממוחשב
ת"י 1080 חלק 28 - בינה מלאכותית - מושגי יסוד ומערכות מומחה
ת"י 1080 חלק 29 - בינה מלאכותית - זיהוי דיבור
ת"י 1080 חלק 30 - בינה מלאכותית – ראיית מחשב
ת"י 1080 חלק 31 - בינה מלאכותית - למידת מכונה
ת"י 1080 חלק 32 - דואר אלקטרוני
ת"י 1080 חלק 33 - היפרמדיה ומולטימדיה
ת"י 1080 חלק 34 - בינה מלאכותית – רשתות עצביות
ת"י 1080 חלק 35 - רישות
ת"י 1080 חלק 36 - למידה מתוקשבת
ת"י 1080 חלק 37 - מציאות מדומה
ת"י 1080 חלק 38[א] - רשתות חברתיות
ת"י 1080 חלק 39[א] - שפות שאילתה

[א] בהכנה.

זיהוי המונחים

כל מונח בתקן מסומן בקוד בן 6 ספרות. זוג הספרות הראשון (משמאל) מציין את מספר החלק בסדרה, זוג הספרות השני מציין את מספר הפרק וזוג הספרות השלישי מציין את מספר הסעיף (הפריט). מספרים אלה זהים בדרך כלל למספרים שבתקן הבין-לאומי.

הוספות והשמטות

לעתים מושמט מונח שקיים במקור, ולעתים כמה מונחים חליפיים באנגלית מתורגמים למונח אחד בעברית.

סוגריים עגולים

מילה או מילים המופיעות בסוגריים עגולים, הן חלק מהמונח אבל ניתן להשמיטן.

סוגריים מרובעים

מילה או מילים המופיעות בסוגריים מרובעים, אינן חלק מהמונח. הן מובאות לשם הבהרה בלבד.

רשימת מונחים

בסוף התקן מובאות שתי רשימות מונחים: לפי סדר א"ב (עברית-אנגלית) ולפי סדר ABC (אנגלית-עברית).

תחום התקן

חלק זה של התקן הישראלי ת"י 1080 מבוסס על התקן הבין-לאומי ISO/IEC 2382-07 מהדורה שנייה שפורסמה בשנת 1999, והוא מביא מונחי תכנות מחשבים.

07.01 - **סוּגֵי שָׂפוֹת** - **Kinds of languages**

07.01.01 **שָׂפָה** **language [in computer programming]**

בתכנות מחשבים, מערכת תווים, כללים ומוסכמות המשמשים להעברת מידע.

07.01.02 **מֵטָא-שָׂפָה** **metalanguage**

שפה המשמשת להגדרת אופיינים, כולם או חלקם, של שפה אחרת, וייתכן גם של עצמה.

07.01.03 **שְׂפַת אַלְגוֹרִיתְמִים** **algorithmic language**

שפה מלאכותית לביטוי אלגוריתמים.

07.01.04 **שְׂפַת תִּכְנוּת** **programming language**

שפה מלאכותית לכתיבת תוכניות מחשב.

07.01.05 **שָׂפָה כְּוֻנַּת-מַחְשֵׁב** **machine-oriented language**
computer-oriented language

שפת תִּכנות שלפסוקים הפשוטים שלה יש מבנה זהה או דומה למבנה של פקודות מכונה של מחשב מסוים או של משפחת מחשבים מסוימת.

07.01.06 **שְׂפַת מְכוֹנָה** **machine language**

שפה מלאכותית המורכבת אך ורק מפקודות מכונה של מחשב מסוים או של משפחת מחשבים מסוימת.

07.01.08 **שְׂפַת-סַף** **assembly language**

שפה כוונת-מחשב, כגון פקודות- מַקרו, המספקת כינוי סימלי לפעולות, למיקומים ולמאפיינים אחרים.

07.01.09 **שְׂפַת דּוֹר רִאשׁוֹן** **first-generation language**
1GL [abb]

שפת תִּכנות הדומה מאד לשפת-סף והתלויה במידה רבה בשפת-מכונה של המחשב.

07.01.10 **שָׂפָה סִמְלִית** **symbolic language**

שפת תִּכנות המכנה פעולות, מענים, אופרנדים ותוצאות בכינויים סמליים.
דוגמות:
שפת-סף, שפת-על.

07.01.11 **שָׂפָה עִלִּית** **high-level language**
שְׂפַת-עָל **high-order language**
HOL [abb]

שפת תִכנות שהתכן והתחביר שלה מוכוונים בעיקר לסוגים מסוימים של בעיות ושאינה תלויה במבנה של מחשב מסוים או של משפחת מחשבים מסוימת.

דוגמות:

Ada, Cobol, Fortran, Pascal.

07.01.12 **שְׂפַת דּוֹר שֵׁנִי** **second-generation language**
2GL [abb]

שפת תִכנות מדור ראשון שהורחבה כך שתכלול מבנים לשוניים של שפת-על, כגון פקודות מַקרו.

07.01.13 **שְׂפַת דּוֹר שְׁלִישִׁי** **third-generation language**
3GL [abb]

שפת-על בעלת יחס גבוה של פקודות מכונה לכל פסוק פשוט שלה, המעלה את רמת ההפשטה של עבודת המתכנת, כדי להתמקד בפתרון הבעיה ולא בידיעה מעמיקה של עבודת מחשב מסוים.

דוגמות: Ada, BASIC, Fortran, Pascal, Modula-2.

07.01.14 **שְׂפַת דּוֹר רְבִיעִי** **fourth-generation language**
4GL [abb]

שפת-על המאפשרת למשתמש, לאו דוקא למתכנת, לכתוב פסוקים בשפה הקרובה לשפה טבעית, שהיא בעלת יחס פקודות מכונה לפסוקים פשוטים, הגבוה בהרבה מהיחס בשפות דור שלישי, והמעלה את רמת ההפשטה של עבודת המשתמש מעבר לרמת ההפשטה של דורות קודמים של שפות תִכנות.

דוגמות:

1. בשפת דור רביעי מיון רשימת לקוחות יבוטא בפסוק כגון:

Sort customer_list on customer_name in ascending order

ללא צורך בידיעה של אלגוריתם מיון כלשהו.

2. dBASE היא שפת דור רביעי.

07.01.16 **שָׂפָה מִתְרַחֶבֶת** **extensible language**

שפת תִכנות הניתנת לשינוי או המשנה את עצמה כדי לספק למתכנת יכולות נוספות המפורטות על ידי המשתמש.

דוגמות:

Ada, C++, FOTRH, LISP, Logo, PROLOG.

07.01.17 **שָׂפָה אַלְגֶּבְּרִית** **algebraic language**

שפת תִּכנות המאפשרת בניית פסוקים הדומים לביטויים אלגבריים.

דוגמות: Ada, Fortran, Pascal.

07.01.18 **שָׂפָה כְּוֻנַּת-בְּעָיָה** **problem-oriented language**
שָׂפָה כְּוֻנַּת-יִשּׁוּם **application-oriented language**

שפת תִּכנות המשקפת את המושגים מתחום יישום מסוים.

דוגמות:

SQL ליישומי מסד נתונים,

COBOL ליישומים מסחריים.

07.01.19 **שָׂפָה כְּוֻנַּת-עֶצֶם** **object-oriented language**

שפת תִּכנות התומכת במושגי תִּכנות כּוון-עצמים.

דוגמות: EIFFEL, Smalltalk.

07.01.20 **שְׂפַת הַכְּרָזָה** **declarative language**

שפת תִּכנות המיועדת להכרזות.

דוגמה:

DDL, שפת הגדרת נתונים.

07.01.21 **שְׂפַת צִוּוּי** **imperative language**

שפת תִּכנות שעיקר ייעודה לשנות מצב משתנים על ידי הצבה.

07.01.22 **שָׂפָה נָהֳלִית** **procedural language**
procedure-oriented language

שפת תִּכנות המספקת את האמצעים לציון התוצאות הנדרשות מפעולות של מערכת עיבוד נתונים, על ידי שימוש בפסוקים מסוימים או בפקודות מסוימות שיש לבצע בסדר מסוים.

דוגמות:

Ada , BASIC, COBOL, Fortran ו- Pascal.

07.01.23 **שָׂפָה לֹא נָהֳלִית** **nonprocedural language**

שפת תִּכנות המספקת את האמצעים לציון התוצאות הנדרשות מפעולות של מערכת עיבוד נתונים, ללא שימוש בפסוקים מסוימים או בפקודות מסוימות שיש לבצע בסדר מסוים.

07.01.24 **שָׂפָה פּוּנְקְצְיוֹנָלִית** **functional language**

שפת תִּכנות המספקת את האמצעים לציון התוצאות הנדרשות מפעולות של מערכת עיבוד נתונים, על ידי שימוש בלעדי בקריאות לפונקציות.

דוגמות:

FORTH, LISP, ML, Miranda, Postcript.

07.01.25 **שְׂפַת תִּכְנוּת מֻבְנֶה** **structured programming language**

שפת תִּכנות המספקת מבנים לשוניים בשביל תִּכנות מובנֶה (2).

07.01.26 **שְׂפַת גּוּשׁ מֻבְנֶה** **block-structured language**

שפת תִּכנות התומכת בשימוש בפסוקי גוש.

דוגמות:

Ada, ALGOL, C, Pascal, PL/1.

07.01.27 **שָׂפָה כְּלָלִית** **general-purpose language**

שפת-על המתאימה לשימוש במִגוון יישומים נרחב.

07.01.28 **שָׂפָה יְעוּדָה** **special-purpose language**

שפת תִּכנות המיועדת לשימוש ביישומים מסוג מסוים.

07.01.50 **שָׂפָה הִדּוּדִית** **interactive language** **conversational language**

שפת תִּכנות התומכת בתקשורת בין משתמש לבין מערכת עיבוד נתונים, על ידי דו-שיח.

07.01.29 **שְׂפַת עִבּוּד רְשִׁימוֹת** **list processing language**

שפת תִּכנות שנועדה לטפלל נתונים המבוטאים בצורת רשימות או מחרוזות תווים.

דוגמה: LISP

07.01.30 **שְׂפַת בִּטּוּיִים** **expression language**

שפת תִּכנות שבה ניתן לבצע הצבות בהקשר לביטוי.

דוגמה: שפת C.

הערה: הביטוי "...if(x=y<0)" חוקי ב-C אך אינו חוקי ב-Ada.

07.01.31 **שְׂפַת תִּסְדּוּר** **text-formatting language**
שְׂפַת תִּסְדּוּר תַּמְלִיל

שפה כוונת-יישום שנועדה לציין את האופן שבו יש לתסדר תמליל.

דוגמות:

HTML, nroff, Hypertext, Markup language

07.01.32 **שְׂפַת סִימְנֵי עֲרִיכָה** **markup language**

שפת תִּסדור תמליל שנועדה להתמיר תמליל גולמי למסמכים מובְנים על ידי תחיבת סימנים נוהליים וסימנים מתארים לתוך התמליל הגולמי.

דוגמה:

SGML היא שפה המיועדת להגדרת שפות התוויה.

07.01.33 **שְׂפַת תֵּאוּר עַמּוּד** **page description language**
PDL [abb]

שפת תִּסדור תמליל המשמשת להגדרת תמונה מוצגת או מודפסת של מסמך, עמוד-עמוד.

דוגמות: HPGL, Postscript.

07.01.34 **שְׂפַת לִמְדָה (לומדה)** **authoring language**

שפה כוונת-יישום המשמשת לפיתוח לומדה להוראה בעזרת מחשב.

07.01.35 **שְׂפַת מַקְרוֹ** (1) **macrolanguage (1)**

שפת תכנות המיועדת להגדרת הגדרות- מַקרו הוראות- מַקרו.

07.01.36 **שְׂפַת מַקְרוֹ** (2) **macrolanguage (2)**

שפת תִכנות הכוללת הגדרות- מַקרו והוראות מַקרו.

07.01.37 **שְׂפַת מִפְרָט** **specification language**

שפה כוונת-יישום, לעִתים קרובות - שילוב של שפה טבעית עם שפה מלאכותית, הניתן לעיבוד במחשב, המשמשת לביטוי דרישות, תכן, התנהגות או אופיינים אחרים של מערכת או של רכיב, והמספקת מבנים לשוניים מיוחדים, ולעיתים גם פרוטוקולי אימות, המשמשים לפיתוח, ניתוח ותיעוד של הישויות המפורטות בשפה.

07.01.38 **שְׂפַת מִפְרַט דְּרִישׁוֹת** **requirement specification language**

שפת מפרט בעלת מבנים לשוניים מיוחדים ולעיתים גם פרוטוקולי אימות, המשמשת לפיתוח, לניתוח ולתיעוד של דרישות חומרה או של דרישות תוכנה או של שתיהן.

07.01.39 **שְׂפַת תֶּכֶן** **design language**

שפת מפרט בעלת מבנים לשוניים מיוחדים ולעיתים גם פרוטוקולי אימות, המשמשת לפיתוח, לניתוח ולתיעוד של התֶּכֶן, של חומרה או של תוכנה.

07.01.40 **שְׂפַת תֶּכֶן חֹמְרָה** **hardware design language**
HDL [abb]

שפת תֶּכֶן בעלת מבנים לשוניים מיוחדים ולעתים גם פרוטוקולים של אימות, המשמשת לפיתוח, לניתוח ולתיעוד של תכן חומרה.

07.01.41 **שְׂפַת תֶּכֶן תָּכְנָה** **program design language**
PDL [abb]

שפת מפרט בעלת מבנים לשוניים מיוחדים ולעיתים גם פרוטוקולי אימות, המשמשת לפיתוח, לניתוח ולתיעוד של תֶּכֶן תוכנה.

07.01.42 **פְּסֵידוֹ-קוֹד** **pseudocode**

צירוף של מבנים לשוניים מתוך שפת תִּכנות עם מבנים לשוניים של שפה טבעית, שאינם בהכרח ניתנים לעיבוד במחשב, המיועדים להבהיר לקורא אנושי את התֶּכֶן של תוכנית

דוגמה:

IF the data arrive faster than expected,
-THEN reject every third input.
-ELSE process all data received.
ENDIF.

07.01.43 **שְׂפַת מִפְרַט מְהַדְּרִים** **compiler specification language**

שפת מפרט המשמשת לפיתוח מהדרים.

07.01.44 **שְׂפַת בְּדִיקָה** **test language**

שפה כוונת-יישום המספקת את האמצעים לבדיקת רכיבים של חומרה או של תוכנה.
דוגמות: ATLAS, ATOLL, DETOLL, DMAD.

07.01.45 **בָּנוּי כַּהֲלָכָה** **well-formed**

מתייחס לתוכנית הכתובה ב**שפת סימני עריכה** ושהתחביר שלה מתאים במלואו לתחביר המחייב של השפה.

07.02 - שִׁיטוֹת, טֶכְנִיקוֹת וּמִבְנֶה - Methods, techniques and structure

07.02.01 **תִּכְנוּת מֻבְנֶה** (1) **structured programming (1)**

שיטה לבניית תוכניות המשתמשת רק במבנים הערוכים מִדרגית כשלכל מבנה בתוכנית יש נקודת כניסה אחת ונקודת יציאה אחת.

הערה:

בתִכנות מובנה משמשים שלושה סוגים של זרימת בקרה: זרימה סִדרתית, זרימה מותנית וזרימה חִזרורית (iterative).

07.02.02 **תִּכְנוּת מֻבְנֶה** (2) **structured programming (2)**

טכניקת פיתוח תוכנה, הכוללת תֶּכֶן מובנה והמביאה לידי פיתוח תוכניות מובְנות.

07.02.03 **תָּכְנִית (תוֹכנית) מֻבְנִית** **structured program**

תוכנית שנבנתה על פי העקרונות של תִכנות מובנה (1).

07.02.04 **תֶּכֶן מֻבְנֶה** **structured design**

כל גישה לתֶכֶן של תוכנה, העומדת בכללים מוגדרים והמבוססת על עקרונות כגון מודולריות, תֶכֶן מעלה-מטה ועידון צעד-צעד של נתונים, של מבני המערכת ושל צעדי העיבוד.

07.02.05 **עִדּוּן צַעַד-צַעַד** **stepwise refinement**

טכניקה לפיתוח תוכנה, שבה מגדירים תחילה באופן כוללני את צעדי העיבוד ואת הנתונים, ואחר כך מגדירים אותם ביתר פירוט.

07.02.06 **קִנֵּן** **nest** [v]

כלל מבנה אחד או יותר מסוג מסוים בתוך מבנה מאותו סוג.

דוגמות:

- קינן לולאה (לולאה מקוננת או פנימית) בתוך לולאה אחרת (לולאה מקַננת או חיצונית).
- קינן שִגרה בתוך שִגרה אחרת.

07.02.07 **תִּכְנוּת פּוּנְקְצִיוֹנָלִי** **functional programming**

שיטה לבניית תוכניות, בעיקר כמעקובת של קריאות לפונקציות, כשהקריאות יכולות להיות גם קריאות מקוננות.

07.02.08 **תִּכְנוּת מוֹדוּלָרִי** **modular programming**

טכניקה לפיתוח תוכנה שבה מפתחים תוכנה כאוסף של מודולים.

07.02.09 **תִּכְנוּת לוֹגִי** **logic programming**

שיטה לבניית תוכניות כקבוצה של כללים לוגיים עם אלגוריתמים מוגדרים מראש לעיבוד נתוני קלט לַתוכנית.

07.02.10 **קְפִיצָה** **jump**

סטייה מביצוע סִדרתי של הוראות התוכנית.

07.02.11 **קָפַץ** **jump** [v]

סטה מביצוע סִדרתי של הוראות התוכנית.

07.02.12 **מַחֲוָן** **indicator**

התקן או משתנה שניתן להציבם במצב מוכתב מראש בהתאם לתוצאות של תהליך או להתרחשות של תנאי מסוים.

דוגמות:

דגל, סמפור.

07.02.13 **דֶּגֶל** **flag**

משתנה המציין מִצָּב (status) של תנאי מסוים.

07.02.14 **מֶתֶג** **switch**

בחירת קפיצה ממבחר קפיצות, הנקבעת על ידי ערכו של דגל.

07.02.15 **שֶׁטַח עֲבוֹדָה** **working space**

work space

working area

work area

חלק מהתקן אִחסון, שהתוכנית משתמשת בו לאִחסון ארעי של נתונים.

07.02.16 **מְנִיעָה הֲדָדִית** **mutual exclusion**

עיקרון הקובע שברגע נתון רק תהליך אסינכרוני אחד רשאי לגשת לאותו משתנה משותף או לבצע אותו קטע קריטי.

07.02.17 **סִנְכְּרוּן** **synchronization**

תיאום הביצוע הבו-זמני של כמה תהליכים אסינכרוניים.

הערה:

ראו סינכרוני, אסינכרוני

07.02.18 **גִּבּוּב** **hashing**

מְעִינָה מְגֻבֶּבֶת **hash addressing**

שיטת חישוב מען על ידי התמרה של מפתח חיפוש כדי לאחסן נתונים במען המחושב או לאחזרם.

07.02.19 **פּוּנְקְצִיַּת גִּבּוּב** **hash function** (in hashing)

פונקצייה המשמשת להחלטה על המיקום של פריט בתוך קבוצת פריטים

הערה:

פונקציית הגיבוב פועלת על שדה המפתח של כל פריט, והתוצאה היא בדרך כלל מיפוי של רבים על אחד..

07.02.20 **מַעַן גִּבּוּב** **hash value**

עֵרֶךְ גִּבּוּב

ערך מספרי המחולָל על ידי פונקציית גיבוב כדי לציין את מיקומו של פריט נתון בהתקן האִחסון.

07.02.21 **הִתְנַגְּשׁוּת גִּבּוּב** **collision** (in hashing)

hash clash

האירוע שבו פונקציית גיבוב מחוללת אותו מען גיבוב לכמה מפתחות שונים.

07.02.22 **פִּתְרוֹן הִתְנַגְּשׁוּת גִּבּוּב** **collision resolution** (in hashing)

גִּבּוּב חוֹזֵר **rehashing**

חישוב נוסף או שימוש בשיטה אחרת לפתרון התנגשות גיבוב.

07.02.23 **מִשְׁתַּנֶּה מְשֻׁתָּף** **shared variable**

משתנה הנגיש לכמה נהלים אסינכרוניים או לכמה תוכניות המתבצעות בו-זמנית.

07.02.24 **כָּרַךְ** **bind** [v]

יִיחס מזהה לעצם בתוכנית.

דוגמה:

ייחס מזהה לערך, למען או למזהה, או סימך פרמטרים פורמליים לפרמטרים ממשיים.

07.02.25 **כְּרִיכָה** **binding**

תהליך הייחוס של מזהה לעצם בתוכנית.

07.02.26 **זְמַן כְּרִיכָה** **binding time**

הרגע שבו הכריכה מתבצעת.

07.02.27 **כְּרִיכָה סְטָטִית** **static binding**

כריכה המתבצעת לפני ביצוע התוכנית ואינה ניתנת לשינוי במהלך ביצוע התוכנית.

07.02.28 **כְּרִיכָה דִּינָמִית** **dynamic binding**

כריכה המתבצעת במהלך ביצוע התוכנית.

07.02.29 **כְּרִיכָה מֻקְדֶּמֶת** **early binding**

תכונה של שפות תִכנות המבצעות בדרך כלל את רוב הכריכות במהלך התרגום, כדי לשפר את יעילות הביצוע של התוכנית.

דוגמות:

Pascal, Fortran, COBOL.

07.02.30 **כְּרִיכָה מְאֻחֶרֶת** **late binding**

תכונה של שפות תִכנות המבצעות את רוב הכריכות במהלך ביצוע התוכנית, בדרך כלל כדי להשיג גמישות בעת הביצוע.

דוגמות:

dBASE, Smalltalk.

07.02.31 **מִצְבּוֹר** **heap**

חלק של האִחסון הפנימי, המשמש לבנייה או למחיקה דינמיות של עצמי נתונים, כאשר סדר השימוש בעצמי הנתונים אינו ניתן לחיזוי.

07.02.32 **זְרִימַת נְתוּנִים** **data flow**

תנועת הנתונים דרך החלקים הפעילים של מערכת עיבוד נתונים, במהלך עבודה מסוימת.

07.03 - חִזְרוּר וּנְסִיגָה - Iteration and recursion

07.03.01 **חִזְרוּר** **iteration**
אִיטֵרַצְיָה
תהליך ביצוע נשנה של מעקובת צעדים.

07.03.02 **צַעַד חִזְרוּר** **iteration step**
ביצוע יחיד של מעקובת צעדי חִזרור.

07.03.03 **לוּלָאָה** **loop**
מעקובת של פסוקים או פקודות שניתן לבצעה
חִזרורית כל עוד קיים תנאי מסוים.

07.03.04 **לוּלָאָה אֵינְסוֹפִית** **infinite loop**
לוּלָאָה סְגוּרָה **closed loop**
לולאה שביצועה נפסק רק על ידי התערבות חיצונית.

07.03.05 **טַעֲנַת תְּנַאי** **loop assertion**
קְבִיעָה
טענה שיש לאמת בכל ביצוע של לולאה.

07.03.06 **גּוּף הַלּוּלָאָה** **loop body**
חלק של לולאה המבצע את מטרתה העיקרית.

07.03.07 **בַּקָּרַת לוּלָאָה** **loop control**
מבנה לשוני הכולל הוראת בדיקה כדי לקבוע אם יש לבצע חִזרור נוסף של לולאה.

07.03.08 **מִשְׁתַּנֶּה בַּקָּרַת לוּלָאָה** **loop-control variable**
פָּרָמֶטֶר לוּלָאָה **loop parameter**
עצם נתונים המשמש כדי לקבוע אם לצאת מתוך לולאה.

07.03.09 **סְכֵמַת חִזְרוּר** **iteration scheme**
השיטה שבה משתמשת בקרת הלולאה כדי לקבוע אם לצאת מתוך לולאה.
דוגמה:
הפסוק "do...while".

07.03.10 **חִזְרוּר קְבוּעַ-מִסְפָּר** **fixed-count iteration**
סכמת חִזרור המסיימת ביצוע לולאה לאחר מספר מוגדר מראש של חִזרורים, ולא עד להתרחשות תנאי מסוים.
דוגמה:
לולאת "do" בשפת Fortran.

07.03.11 **מִבְחַן סִיּוּם** **termination test**
בבקרת לולאה, הבדיקה שבה התוצאה "אמת" מורה על סיום החִזרור.

07.03.12 **מִבְחַן הֶמְשֵׁךְ** **continuation test**
בבקרת לולאה, הבדיקה שבה התוצאה "אמת" מורה על המשך החִזרורים ואילו התוצאה "שקר" מורה על סיום החִזרור.

07.03.13 **מִבְחַן-קֶדֶם שֶׁל לוּלָאָה** **pretest loop**

בקרת לולאה המבצעת בדיקה לפני כניסה לגוף הלולאה.

דוגמה:

לולאת "for" בשפת Ada.

07.03.14 **מִבְחָן לְאַחַר לוּלָאָה** **posttest loop**

בקרת לולאה המבצעת בדיקה לאחר ביצוע גוף הלולאה.

דוגמה:

המבנה "repeat ... until" ב-Pascal.

07.03.15 **מִבְחַן-תָּוֶךְ שֶׁל לוּלָאָה** **in-test loop**

בקרת לולאה המבצעת בדיקה במקום כלשהו בתוך גוף הלולאה.

דוגמה:

הפסוק "exit" בשפת Ada.

07.03.16 **נְסִיגָה** **recursion**

רֶקוּרְסִיָה

תהליך שבו שגרה כוללת קריאה לעצמה או קריאה לשגרה אחרת הקוראת לשגרה המקורית, או מאתחלת שרשרת קריאות לשגרות, המובילה לבסוף לקריאה חוזרת לשגרה המקורית.

07.03.17 **נָסוֹג יְשִׁירוֹת** **directly recursive**

מתייחס לשגרה הכוללת קריאה לעצמה.

07.03.18 **נָסוֹג בַּעֲקִיפִין** **indirectly recursive**

מתייחס לשגרה הקוראת לשגרה אחרת הקוראת לַשגרה המקורית או המאתחלת שרשרת קריאות לשגרות, המובילה לבסוף לקריאה חוזרת לשגרה המקורית.

07.03.19 **נְסִיגָה הֲדָדִית** **simultaneous recursion**

מצב שבו שתי שגרות קוראות זו לזו.

07.03.20 **רַב-כְּנִיסוֹת** **reentrant**

מתייחס לתוכנית או לחלק של תוכנית, שיכול לשכון בזכרון כעותק משותף אחד עבור כל התהליכונים היכולים להיכנס לתוכו חזור והיכנס בזמן הביצוע, אפילו לפני סיום הביצוע הקודם, וכל ביצוע של תוכנית כזו אינו תלוי בביצועים אחרים של אותה תוכנית.

הערה:

קוד רב-כניסות לא משנה את עצמו ומספק שטחי זכרון נפרדים לכל תהליכון.

07.04 - הֲכָנַת תָּכְנִיּוֹת - Program preparation

07.04.01 **מְתַכְנֵת** **programmer**
תָּכְנִיתָן (**תוכניתן**)
תַּכְנָת
אדם העוסק בתכן, בכתיבה או בבדיקה של תוכניות.

07.04.02 **סְבִיבָה** **environment**
אוסף כלי תוכנה וחומרה התומכים בשלב אחד או יותר של מחזור פיתוח התוכנה.

07.04.03 **סְבִיבַת תִּכְנוּת** **programming environment**
programming support environment
אוסף כלי תוכנה וחומרה התומכים בהכנת תוכניות.

07.04.04 **סְבִיבַת תִּכְנוּת כִּלּוּלִית** **integrated programming environment**
IPE [abb]
אוסף כִּילוּלִי של כלי תוכנה וחומרה עם מִנשָק-משתמש אחיד (בדרך כלל גרפי), התומכים בפיתוח תוכניות.

07.04.05 **תִּרְגֵּם** **translate** [v]
הִתמיר תוכנית הכתובה בשפת תִכנות אחת לשפת תִכנות אחרת ללא שינוי המשמעות המקורית.

07.04.06 **תִּרְגוּם** **translation**
פעולת התִרגום או תוצאתה.

07.04.07 **מְתַרְגֵּם** [שם עצם] **translator**
תָּכְנִית (**תוכנית**) **תִּרְגוּם** **translation program**
תוכנית לתרגום.

07 04.08 **אִסֵּף** **assemble** [v]
תִרגם לשפת יעד תוכנית כתובה בשפת-סף.

07.04.09 **מְאַסֵּף** **assembler**
תוכנית תִרגום לאיסוף.

07.04.10 **מְאַסֵּף מֻחְלָט** **absolute assembler**
מאסף המפיק קוד מוחלט.

07.04.11 **קוֹד** **code** [in computer programming]
בתִכנות מחשבים, חלק של תמליל תוכנית, הכתוב בשפת תִכנות או בצורה המופקת על ידי מאסף, מהדר או מתרגם אחר.

07.04.12 **קִדּוּד** **coding** [in computer programming]

בתִכנות מחשבים, תהליך כתיבת תוכנית בשפת תִכנות.

07.04.13 **קוֹד מֻחְלָט** **absolute code**

קוד שבו כל מען הוא מען מוחלט.

07.04.14 **קוֹד מְאַסֵּף** **assembly code**

קוד הניתן לעיבוד על ידי מאסף.

07.04.15 **בְּסִיס-סַף** **assembled origin**

המען של המיקום התחילי בזיכרון, שהוקצה לתוכנית, כולה או חלקה, על ידי מאסף, מהדר או תוכנית קישור.

07.04.16 **מְאַסֵּף צוֹלֵב** **cross-assembler**

מאסף המשתמש במחשב מסוג אחד כדי לאסף תוכנית לשפת יעד של מחשב מסוג אחר.

07.04.17 **מְאַסֵּף הֲזָחָה** **relocating assembler**

מאסף המפיק תוכנית הניתנת למיקום מחדש.

07.04.18 **אַסֵּף וּבַצֵּעַ** **assemble-and-go**

שיטת תִפעול שאין בה עצירות בין האיסוף, הקישור, הטעינה והביצוע של תוכנית.

07.04.19 **הִדֵּר** **compile** [v]

תִרגם תוכנית, כולה או חלקה, הכתובה בשפת-על לתוכנית הכתובה בשפת-ביניים, בשפת-סף או בשפת-מכונה.

07.04.20 **מְהַדֵּר** **compiler**

תוכנית תִרגום המבצעת הידור.

07.04.21 **הִדּוּר** **compilation**

התהליך או התוצאה של פעולת המהדר.

07.04.22 **יְחִידַת הִדּוּר** **compilation unit**

תוכנית, כולה או חלקה, הכתובה בשפת-על, שהיא שלמה במידה מספקת להידור.

07.04.23 **קוֹד מְהַדֵּר** **compiler code**

קוד כתוב בצורה המוכרת למהדר והניתנת לזיהוי ולעיבוד על ידיו.

מְחוֹלֵל מְהַדְּרִים **compiler generator**

07.04.24 **compiler compiler**
metacompiler

מתרגם או מפרש המשמשים להגדרה ולבנייה של מהדר.

07.04.25 **מְהַדֵּר צוֹלֵב** **cross-compiler**

מהדר המהדר תוכנית במחשב אחד כדי להדר תוכניות לשפת יעד של מחשב מסוג שונה.

07.04.26 **הַדֵּר וּבַצֵּעַ** **compile-and-go**

שיטת תִפעול שאין בה עצירות בין ההידור, הקישור, הטעינה והביצוע של תוכנית.

07.04.27 **שַׁחְזֵר לִשְׂפַת-סַף** **disassemble** [v]

תִרגם קוד יעד לשפת-סף.

07.04.28 **שִׁחְזֵר לִשְׁפַת-עָל** **decompile** [v]

תִּרגם תוכנית מהודרת מהגירסה שלה בשפת-מכונה לצורה הדומה לתוכנית המקורית בשפת-על.

07.04.29 **תָּכְנִית (תוֹכנית) שִׁחְזוּר לִשְׁפַת-עָל** **decompiler**

כלי תוכנה המשחזר תוכניות לשפת-על.

07.04.30 **פֵּרֵשׁ** **interpret** [v]

ניתח, תִּרגם וביצע כל פסוק או מבנה בתוכנית מקור לפני הטיפול בפסוק הבא.

07.04.31 **מְפָרֵשׁ** **interpreter**

תָּכְנִית (תוכנית) פֵּירוּשׁ **interpretive program**

תוכנית תִּרגום המסוגלת לפרש.

07.04.32 **קוֹד פֵּירוּשׁ** **interpretive code**

קוד הכתוב בצורה הניתנת לזיהוי ולעיבוד על ידי מפרש.

07.04.33 **קוֹד מְכוֹנָה** **machine code**

קוד הכתוב בצורה הניתנת לזיהוי ולעיבוד על ידי יחידת העיבוד של מחשב.

07.04.34 **שְׂפַת מָקוֹר** **source language**

שפת תְּכנות המשמשת לכתיבת תוכנית מקור.

07.04.35 **תְּלוּי-מְכוֹנָה** **machine-dependent**

מתייחס לתוכנה המסתמכת על תכונות ייחודיות של סוג מסוים של מחשבים וניתנת לפיכך לביצוע אך ורק במחשבים מאותו סוג.

07.04.36 **לֹא תְּלוּי-מְכוֹנָה** **machine-independent**

מתייחס לתוכנה שאינה מסתמכת על תכונות ייחודיות של סוג מסוים של מחשבים וניתנת לפיכך לביצוע במחשבים מסוגים שונים.

07.04.37 **תָּכְנִית (תוכנית) מָקוֹר** **source program**

תוכנית הניתנת לתרגום על ידי מתרגם מסוים.

07.04.38 **קוֹד מָקוֹר** **source code**

קוד הכתוב בצורה המתאימה להיות קלט למאסף, למהדר או למתרגם אחר.

07.04.39 **מוֹדוּל מָקוֹר** **source module**

תוכנית מקור, כולה או חלקה, שהיא שלמה במידה מספקת לתִּרגום.

07.04.40 **שְׂפַת בֵּינַיִם** **intermediate language**

שפת יעד שמתרגמים אליה משפת מקור תוכנית מקור, כולה או חלקה, או פסוק בודד, וממשיכים לתרגם או לפרש אותה.

הערה:

לצורך המשך התרגום, שפת הביניים עשויה לשמש שפת מקור.

07.04.41 **מְהַדֵּר שׁוֹרֶשׁ** **root compiler**

מהדר המהדר לשפת ביניים בלבד.

הערה:

מהדר שורש בצירוף עם מחולל קוד מהווה מהדר מלא.

07.04.42 **מְחוֹלֵל קוֹד** **code generator**

שגרה, לעיתים חלק ממהדר, המתמירה תוכנית, כולה או חלקה, משפת ביניים לשפת יעד.

07.04.43 **מְחוֹלֵל קוֹד-מָקוֹר** **source code generator**

כלי תוכנה הקולט את הדרישות או את התכן של תוכנית ומייצר קוד מקור הממ‌מש את הדרישות או את התכן.

07.04.44 **פִּרֵק** **parse** [v]

החליט על המבנה התחבירי של מבנה לשוני על ידי ניתוחו ליחידות לשוניות וקביעת היוחסות (relationships)ביניהם.

דוגמה:

פירק גוש תמליל לפסוקים, פסוקים לביטויים וביטויים לפעולות ואופרנדים.

07.04.45 **מְפָרֵק** **parser**

נַתָּח תַּחְבִּירִי

כלי תוכנה המנתח תוכניות או תמליל אחר, לעתים קרובות כצעד ראשון של איסוף, הידור, פירוש או ניתוח.

07.04.46 **מְחוֹלֵל יִשּׁוּמִים** **application generator**

מחולל קודי מקור, המייצר תוכניות לפתרון בעיות בתחום יישום מסוים.

07.04.47 **כְּלִי תָּכְנָה (תוכנה)** **software tool**

תוכנה המשמשת לפיתוח, לבדיקה, לניתוח, לתִחזוק או לתיעוד של תוכניות.

דוגמות:

מחולל התייחסויות הדדיות, תוכנית שִחזור לשפת-על, תוכנית הינע, עורך תמלילים, עורך תרשימי זרימה, מוניטור, מחולל אירועי בדיקה, נַתָּח תִזמון.

07.04.48 **שְׂפַת מַטָּרָה** **target language**

שפה שבה המתרגם מבטא את תוצאות התרגום.

07.04.49 **מַחְשֵׁב מַטָּרָה** (1) **target machine (1)**

המחשב שתוכנית מיועדת להתבצע בו.

הערה:

ראו מחשב מארח (1).

07.04.50 **מַחְשֵׁב מַטָּרָה** (2) **target machine (2)**

מחשב שמחשב אחר מחקה אותו.

הערה:

ראו מחשב מארח (2).

07.04.51 **תָּכְנִית (תוכנית) מַטָּרָה** **target program**

הגירסה המתורגמת של תוכנית מקור.

07.04.52 **שָׂפָה מְאָרַחַת** **host language**

שפת תִכנות שמשוכנים בה פסוקים של שפת טִפלוּל נתונים.

07.04.53 **מַחְשֵׁב מְאָרֵחַ** (1) **host machine (1)**

מחשב המשמש לפיתוח תוכנה המיועדת למחשב אחר.

הערה:

ראו מחשב מטרה (1).

07.04.54 **מַחְשֵׁב מְאָרֵחַ** (2) **host machine (2)**

מחשב המשמש לחיקוי של מחשב אחר.

הערה:

ראו: מחשב מטרה (2).

07.04.55 **מַחְשֵׁב מְאָרֵחַ** (3) **host machine (3)**

המחשב שתוכנית או קובץ מותקנים בו.

07.04.56 **שְׂפַת יַעַד** **object language**

שפת מטרה לביטוי תוכניות יעד.

07.04.57 **קוֹד יַעַד** **object code**

קוד הנוצר על ידי מאסף או מהדר.

הערה:

תוכנית יעד עשויה מקוד יעד.

07.04.58 **מוֹדוּל יַעַד** **object module**

תוכנית יעד, כולה או חלקה, שלמה במידה

הערה:

מהדרים ומאספים מייצרים בדרך כלל מודול יעד.מספקת לצורך קישור.

07.04.59 **תָּכְנִית (תוכנית) יַעַד** **object program**

תוכנית מטרה הניתנת לביצוע במחשב מסוים לאחר קישור.

הערה:

התוכנית טעונה קישור לפי הצורך.

07.04.60 **זְמַן תִּרְגּוּם** (1) **translation time (1)**

כל רגע במהלך תִרגום.

07.04.61 **זְמַן הִדּוּר** (1) **compilation time (1)**

כל רגע במהלך הידור.

07.04.62 **זְמַן אִסּוּף** (1) **assembly time (1)**

כל רגע במהלך איסוף.

07.04.63 **מֶשֶׁךְ תִּרְגּוּם** **translation duration**
translation time (2)

משך הזמן הנדרש לתִרגום תוכנית.

07.04.64 **מֶשֶׁךְ הִדּוּר** **compilation duration**
compilation time (2)

משך הזמן הנדרש להידור תוכנית.

07.04.65 **מֶשֶׁךְ אִסּוּף** **assembly duration**
assembly time (2)

משך הזמן הנדרש לאיסוף תוכנית.

07.04.66 **הַנְחָיַת מְתַרְגֵּם** **translator directive**

מבנה לשוני המשמש לבקרה של תרגום תוכנית.

07.04.67 **הַנְחָיַת מְאַסֵּף** **assembler directive**

מבנה לשוני המשמש לבקרה של איסוף תוכנית.

07.04.68 **הַנְחָיַת מְהַדֵּר** **compiler directive**

מבנה לשוני המשמש לבקרה של הידור תוכנית.

07.04.69 **הַנְחָיַת מְפָרֵשׁ** **interpreter directive**

מבנה לשוני המשמש לבקרה של פירוש תוכנית.

07.04.70 **הִדּוּר תָּלוּי** **separate compilation**

dependent compilation

הידור נפרד של מודול מקור, המשתמש בנתונים המייצגים יוחסות של מִנשָׂקים והֶקשרים ממודולי מָקור מסומכים.

הערה:

נתוני מִנשקים והֶקשרים משמשים למהדר לבידוק תקֵפוּת ולפתרון התייחסויות.

07.04.71 **הִדּוּר לֹא-תָּלוּי** **independent compilation**

הידור נפרד של מודול מקור, שאינו משתמש בנתונים המייצגים יוחסות של מִנשָׂקים והֶקשרים ממודולי מָקור מסומכים.

הערה:

כאשר מודולים שהודרו באופן לא-תלוי מצורפים בסופו של דבר יחד, עלול להיווצר הצורך לבידוק תקֵפוּת של נתוני המִנשקים והֶקשרים ביניהם.

07.04.72 **יְחִידָה גֶּנֶרִית** **generic unit**

יְחִידַת יְסוֹד

מודל של מבנה לשוני, ייתכן פרמטרי, שממנו נגזר מבנה לשוני תקין בזמן התרגום.

07.04.73 **מְחוֹלֵל מַקְרוֹ** **macrogenerator**

מודול, בדרך כלל חלק ממאסף או מהדר, המחליף כל פקודת מַקרו או קריאת מַקרו בתוכנית מקור, בקוד מתאים לפי הגדרות המַקרו המתאימות.

07.04.74 **מְעַבֵּד מַקְרוֹ** **macroprocessor**

שגרה הכלולה בכמה מאספים ומהדרים לתמיכה בהגדרת מַקרו.

07.04.75 **תִּכְנוּת מַקְרוֹ** **macroprogramming**

תִכנות המשתמש בהגדרות מַקרו, בפקודות מַקרו או בקריאות מַקרו.

07.04.76 **סִפְרִיַּת מַקְרוֹ** **macro library**

אוסף של קריאות- מַקרו ופקודות- מַקרו לרבות הגדרות- המַקרו שלהן, הזָמין לשימוש על ידי מחולל- מַקרו.

07.04.77 **מְאַסֵּף מַקְרוֹ** **macroassembler**

מאסף הכולל או מבצע את התִפקודים של מחולל מַקרו.

07.04.78 **מְחוֹלֵל תָּכְנִיּוֹת (תוכניות)** **program generator**

תוכנית היכולה לייצר תוכניות אחרות.

07.04.79 **מְעַבֵּד קְדָם** **preprocessor**

תוכנית או שגרה המבצעות כמה צעדי עיבוד לפני עיבוד עיקרי.

07.04.80 **עִבּוּד קְדָם** **preprocessing**

עיבוד המתבצע לפני העיבוד העיקרי.

07.04.81 **בַּר-עָבָר** **stateful**

מתייחס לתוכנית, מחשב או ישות אחרת שעוקבים אחר מהלך האירועים בתהליך נתון, זוכרים אותם ומאפשרים חזרה על אירועים מן העבר.

07.04.82 **סַר-עָבָר** **stateless**

מתייחס לתוכנית, מחשב או ישות אחרת שלא עוקבים אחר מהלך האירועים, לא זוכרים אותם ולא מאפשרים חזרה על אירועים מן העבר.

07.04.83 **תִּכְנוּת סַר-עָבָר** **stateless programming**

תכנות חסר רכיבים שעוקבים אחר מהלך האירועים בתוכנית וזוכרים אותם, ומשום כך אינו מאפשר לחזור על אירועי התוכנית.

07.04.84 **מַשְׁאַב בַּר-עָבָר** **stateful resource**

משאב הכולל רכיבים שעוקבים אחר מהלך האירועים בתוכנית וזוכרים אותם, ומשום כך מאפשר לחזור על אירועי התוכנית.

07.04.85 **שֶׁרֶת סַר-עָבָר** **stateless server**

תכנות חסר רכיבים שעוקבים אחר מהלך האירועים בשרת וזוכרים אותם, ומשום כך אינו מאפשר לחזור על האירועים בשרת.

07.04.86 **תַּרְמִיל סַר-עָבָר** **stateless bean**

שגרה בתוכנית החסרה רכיבים שעוקבים אחר מהלך האירועים בתוכנית וזוכרים אותם, ומשום כך אינה מאפשרת לחזור על אירועי התוכנית.

07.05 - קִשּׁוּר וּטְעִינָה - Linking and loading

07.05.01 **קִשֵּׁר** **link** [v]

חיברר עצמי נתונים על ידי הצבת מצביעים, או חיברר חלקים של תוכנית אחת או יותר על ידי יצירת קישורים.

דוגמה: קישר תוכניות יעד על ידי עורך קישור.

07.05.02 **קִשּׁוּר** [שם עצם] **link**

linkage

חלק של תוכנית, לעִתים פקודה או מען בודדים, המעביר בקרה, וייתכן גם פרמטרים, בין מודולים נפרדים של תוכנית זו.

07.05.03 **עוֹרֵךְ קִשּׁוּרִים** **linkage editor**

מְקַשֵּׁר **linker**

תוכנית המעבדת כמה מודולי יעד או מודולי טעינה שתורגמו בצורה לא-תלויה, פותרת הפניות צולבות ביניהם, מקשרת ביניהם, בונה אלמנטים זחיחים, ובמידת הצורך מתאימה מענים, ועל ידי כך יוצרת מודולי טעינה חדשים.

07.05.04 **מַטְעֵן** **loader**

תוכנית המעתיקה תוכניות אחרות מאחסון חיצוני לזיכרון פנימי של המחשב או המעתיקה נתונים מאחסון חיצוני לזיכרון פנימי או מזיכרון פנימי לאוגרים.

07.05.05 **טָעַן** **load** [v] [in computer programming]

בתכנות מחשבים, הֵריץ מַטעֵן.

07.05.06 **מַטְעֵן מֻחְלָט** **absolute loader**

תוכנית המעתיקה מודולי טעינה, שבה כל המענים הם מענים מוחלטים, מאחסון חיצוני לזיכרון פנימי, כך שאין צורך בהתאמת מענים.

07.05.07 **מַטְעֵן מְקַשֵּׁר** **linking loader**

תוכנית המשלבת את היכולות התִפקודיות של עורך קישור ושל מַטעֵן.

07.05.08 **מוֹדוּל טְעִינָה** **load module**

תוכנית, כולה או חלקה, המתאימה לטעינה ולביצוע.

הערה:

מודול טעינה הוא בדרך כלל תוצאה של עורך קישור.

07.05.09 **טְעַן וּבַצַּע** **load-and-go**

שיטת פעולה שאין בה עצירות בין טעינת התוכנית לביצועה.

07.05.10 **בְּסִיס טְעִינָה** **loaded origin**

מען מקום האחסון התחילי של תוכנית שנטענה לתוך הזיכרון הראשי של מחשב.

07.05.11 **מַפַּת טְעִינָה** **load map**

רשימה שנוצרה על ידי המחשב, המזהה את המיקום בזיכרון או את הגדלים של כל התוכניות או של חלקים נבחרים שלהן או של נתונים הנמצאים בזיכרון.

07.05.12 **הִזִּיחַ** **relocate** [v]

העביר תוכנית, כולה או חלקה, ממקום אחד למקום אחר במרחב המענים, ועשה את ההתאמות הדרושות במענים כדי שהחלקים התואמים של התוכנית יוכלו להתבצע במקומות החדשים.

07.05.13 **תָּכְנִית (תוכנית) זְחִיחָה** **relocatable program**

תוכנית שאפשר להזיחה.

07.05.14 **זָחִיחַ** **relocatable**

מתייחס לתוכנית, כולה או חלקה, שאפשר לטעון אותה לתוך כל מקום בזיכרון הראשי.

הערה:

מען בסיס הטעינה נקבע על ידי המַטעֵן המתאים את המענים כך שישקפו את המקומות בזיכרון, שלתוכם נטענו חלקי התוכנית.

07.05.15 **מַטְעֵן זָחִיחַ** **relocating loader**

מטען המטפל בתוכניות זחיחות או במודולים זחיחים.

07.05.16 **מַדְרִיךְ הֲזָחָה** **relocation dictionary**

חלק של מודול יעד או של מודול טעינה, המזהה את המענים שיש לתַאם בעת הזחה.

07.05.17 **הֶסֵּט הֲזָחָה** **relocation offset**

ההפרש בין בסיס-טעינה לבין בסיס-סף של תוכנית.

07.05.18 **הֶסֵּט מַעַן** **address offset**

מספר שיש להוסיף למען יחסי כדי לקבוע את מענו של המיקום בזיכרון שאליו מבקשים לגשת.

07.05.19 **הַקְטָעָה** **segmentation**

טכניקה להקצאת שטחי זיכרון ראשי כדי לטעון לתוכו חלקי תוכנית מאִחסון עזר לפי הצורך.

07.05.20 **מִקְטָע** **segment** [in computer programming]

בתִכנות מחשבים, חלק של תוכנית הניתן לביצוע בלי שהתוכנית כולה נמצאת בזיכרון הראשי.

07.05.21 **מִקְטַע רׄבֶד** **overlay segment**

כל מִקטע מתוך כמה מקטעים של תוכנית, שכדי להתבצע תופס בתורו אותו שטח של הזיכרון הראשי.

07.05.22 **רִבֵּד** **overlay** [v]

טען מִקטע רובד מאִחסון עזר במקום רובד אחר.

07.05.23 **מְפַקֵּחַ רִבָּדִים** **overlay supervisor**

שגרה המבקרת את סדר הטעינה ואת ההצבה של מקטעי רובד.

07.05.24 **תּוֹשָׁב** [שם תואר] **resident** [adj]

מתייחס לתוכניות, לחלקי תוכניות או לנתונים, כל עוד הם שוכנים בזיכרון הראשי.

07.05.25 **תָּכְנִית (תוכנית) תּוֹשֶׁבֶת** **resident program**

תוכנית השוכנת בשטח אִחסון מסוים.

07.05.26 **רְשׁוּמַת הַפְּעָלָה** **activation record**

עצם הנתונים (the dada object) הכולל מידע המייצג הֶדגם (instance) של מטלה או שגרה, והכולל ערכי נתונים ונתוני מצב עיבוד עבור אותו הדגם.

הערה:

רשומת הפעלה עשויה לכלול פרמטרים, תוצאות, נתונים מקומיים וכדומה.

07.06 - בְּצוּעַ תָּכְנִית - Program execution

07.06.01 **מְעַבֵּד שָׂפָה** **language processor**

יחידה תִפקודית המתרגמת ומבצעת תוכניות הכתובות בשפת תִכנות מסוימת.

דוגמה:

מכונת **LISP**.

07.06.02 **זְמַן בְּצוּעַ** (1) **execution time (1)**

זְמַן רִיצָה (1) **run time (1)**

כל רגע שבו מתבצעת תוכנית מסוימת.

07.06.03 **מֶשֶׁךְ בְּצוּעַ** **execution time (2)**

run time (2)

running time

משך הזמן הנדרש לביצוע תוכנית מסוימת.

הערה:

זמן ביצוע יכול להיות זמן שעון או זמן מעבד.

07.06.04 **זְמַן שָׁעוֹן** **elapsed time**

wall-clock time

משך הזמן הממשי שחלף מהתחלת ביצוע תוכנית ועד לַסיום.

הערה:

בשונה מזמן מעבד.

07.06.05 **זְמַן מְעַבֵּד** **processor time**

סכום פרקי הזמן שבהם מעבד ביצע בפועל את התוכנית.

הערה:

בשונה מזמן שעון.

07.06.06 **פְּרוֹפִיל בְּצוּעַ** **execution profile**

ייצוג מוחלט או יחסי של שכיחויות ביצוע או של זמני ביצוע (2), של פקודות או של פסוקים של תוכנית.

07.06.07 **עִקְבָה** **trace**

רשומת הביצוע של תוכנית, כולה או חלקה, המראה את מעקובת הפקודות או הפסוקים שבוצעו, את האופרנדים המעורבים ושמותיהם, ואת התוצאות.

07.06.08 **עָקַב** **trace** [v]

הכין רשומת מעקב (עִקבָה).

07.06.09 **עִקְבַת בְּצוּעַ** **execution trace**

control-flow trace

code trace /US/

רשומת מעקובת של פקודות שבוצעו במהלך הביצוע של תוכנית.

07.06.10 **עִקְבָה הִיסְטוֹרִית** **retrospective trace**

עִקבָה המיוצרת לאחר ביצוע תוכנית, מתוך נתונים היסטוריים שנרשמו במהלך הביצוע.

הערה:

בשונה מרשומת מעקב של ביצוע המתבצעת באופן מצטבר במהלך ביצוע התוכנית.

07.06.11 **עִקְבַת שְׁגָרוֹת** **subprogram trace**

עִקבָה של קריאות, כולן או חלקן, לשגרות שבוצעו במהלך ביצוע תוכנית, כולה או חלקה, ואולי גם של ערכי הפרמטרים שהועברו לכל שגרה או מודול ושהוחזרו מהם.

07.06.12 **עִקְבַת סְמָלִים** **symbolic trace**

עִקבָה של הפסוקים בתוכנית המקור ושל תוצאות קפיצות בעת ביצוע התוכנית, המשתמשת בעיקר בסמלים ולא בערכים ממשיים.

07.06.13 **בִּצּוּעַ סִמְלִי** **symbolic execution**

תהליך התומך בניתוח תוכנה על ידי הדמיה של ביצוע תוכנית, כולה או חלקה, תוך כדי שימוש בסמלים כנתוני קלט, כגון שמות של משתנים כתחליף לערכים הממשיים שלהם, וביטוי של פלט התוכנית כביטויים מתמטיים או לוגיים שבהם מופיעים סמלים אלה.

07.06.14 **עִקְבַת מִשְׁתַּנִּים** **variable trace**

עִקְבַת נְתוּנִים **data-flow trace**

data trace

עִקבָה של השמות והערכים של משתנים, שנגשו אליהם או ששונו במהלך ביצוע תוכנית.

07.06.15 **מוֹנִיטוֹר בִּצּוּעַ** **execution monitor**

כלי תוכנה או התקן חומרה הפועלים בו-זמנית עם מערכת או יחידה תִפקודית, והמפקחים, רושמים, מנתחים או מאמתים את התפעול של המערכת או של היחידה התפקודית.

07.06.16 **יָצָא** **exit** [v]

ביצע פקודה או פסוק בתוכנית או בחלק של תוכנית, הגורמים להפסקת הביצוע של התוכנית או של חלקה.

07.06.17 **נְקֻדַּת יְצִיאָה** **exit point**

הנקודה בתוכנית, במודול או בפסוק, שבה ניתן לסיים את הביצוע של התוכנית, המודול או הפסוק.

07.06.18 **נְקֻדַּת כְּנִיסָה** **entry point**

כְּנִיסָה **entrance**

הנקודה בתוכנית, במודול או בפסוק שבה ניתן להתחיל את הביצוע של התוכנית, המודול או הפסוק.

07.06.19 **נְקֻדַּת חֲזָרָה** **reentry point**

הנקודה בתוכנית, במודול או בפסוק, שבה הם מחדשים את ביצועם לאחר הביצוע של תוכנית, מודול או פסוק אחרים.

07.06.20 **מִפְסָק** **breakpoint**

נקודה בתוכנית, במודול או בפסוק, שבה ניתן להפסיק את הביצוע עקב מצב או אירוע מסוימים.

הערות:

1. מציבים מִפְסָק כדי לאפשר ניטור ידני או אוטומטי של ביצועי תוכנית או תוצאותיה.
2. יתכן יותר ממפסק אחד.

07.06.21 **הִצִיב מִפְסָק** **set a breakpoint (v)**

הגדיר מפסק וכן אירוע שיגרום הפסקת ביצוע של תוכנית.

07.06.22 **הִפְסִיק** **initiate a breakpoint (v)**

גרם הפסקת ביצוע של תוכנית במִפסָק.

07.06.23 **מִפְסַק קוֹד** **code breakpoint**

control breakpoint

מִפסק התלוי בביצוע פקודה מסוימת.

07.06.24 **מִפְסַק נְתוּנִים** **data breakpoint**

מִפסק התלוי בגישה לעצם נתונים מסוים.

07.06.25 **מִפְסָק דִּינָמִי** **dynamic breakpoint**

נקודה בתוכנית או בפסוק, שבה ניתן להפסיק את הביצוע עקב מצב או אירוע מסוימים, המשתנים במהלך ביצוע תוכנית זו או אחרת.

07.06.26 **מִפְסָק סְטָטִי** **static breakpoint**

מפסק המוצב במהלך הידור תוכנית.

07.06.27 **מִפְסַק תָּכְנִית (תוכנית) נִפּוּי** **programmable breakpoint**

מִפסק הגורם הפעלה אוטומטית של תוכנית ניפוי מוגדרת מראש.

07.06.28 **מִפְסַק כְּנִיסָה** **preamble breakpoint**

מִפסק הממוקם בנקודת הכניסה של תוכנית או של שגרה.

07.06.29 **מִפְסַק יְצִיאָה** **postamble breakpoint**

מִפסק הממוקם בנקודת היציאה מתוכנית או שגרה.

07.06.30 **נְקֻדַּת בִּקֹּרֶת** **checkpoint**

נקודה בתוכנית, המתאימה להפסקת ביצוע התוכנית, שבה הוכנסה מעקובת פקודות כדי לרשום את המצב והתוצאות, לבחון אותם ולשתחל את התוכנית.

07.06.31 **שִׁתְחֵל** **restart** [v]

חידש ביצוע תוכנית על ידי שימוש בנתונים שנרשמו בנקודת ביקורת.

07.06.32 **נְקֻדַּת שִׁתְחוּל** **restart point**

rescue point

נקודה בתוכנית שבה ניתן להמשיך בביצוע התוכנית או לחדש את הביצוע לאחר שהופסק במִפסק או בנקודת ביקורת.

07.06.33 **הִתְאוֹשֵׁשׁ** **recover** [v]

קבע מצב קודם או מצב חדש של מערכת, קובץ, מסד נתונים או משאב אחר, כולם או חלקם, או של ביצוע תוכנית, כך שניתן עתה לבצע תִפקודים דרושים.

07.06.34 **הִתְאוֹשְׁשׁוּת** **recovery** [in computer programming]

בתכנות מחשבים, קביעת מצב קודם או מצב חדש של מערכת, קובץ, מסד נתונים או משאב אחר, כולם או חלקם, או של ביצוע תוכנית, כך שניתן לבצע תִּפקודים דרושים.

07.06.35 **הִתְאוֹשְׁשׁוּת לְפָנִים** **forward recovery**

הִתְאוֹשְׁשׁוּת קְדוֹמַנִּית

התאוששות שבה מערכת, תוכנית, קובץ, מסד נתונים או משאב אחר מובאים למצב חדש השונה מהקודם, שבו יוכלו לבצע תִּפקודים דרושים.

דוגמה:

שחזור קובץ למצב נתון על ידי עדכון גִרסה קודמת של הקובץ, בנתונים שנרשמו בסדר הכרונולוגי של השינויים בקובץ.

07.06.36 **הִתְאוֹשְׁשׁוּת לְאָחוֹר** **backward recovery**

הִתְאוֹשְׁשׁוּת אֲחוֹרַנִּית

התאוששות שבה מערכת, תוכנית, קובץ, מסד נתונים או משאב אחר, משוחזרים למצב קודם שבו יוכלו לבצע תִּפקודים דרושים.

דוגמה:

שחזור של קובץ למצב נתון על ידי ביטול כל השינויים שבוצעו בו מאז שהיה במצב זה.

07.06.37 **הִתְאוֹשְׁשׁוּת יְשִׁירָה** **inline recovery**

התאוששות המתבצעת על ידי חידוש העבודה בנקודה הקודמת לַכָּשֶׁל.

07.06.38 **הַרְעָבָה** **starvation**

מצב שבו תהליך אסינכרוני אינו מסוגל להתקדם בתוך פרק זמן הניתן לחיזוי מפני שתהליכים אסינכרוניים אחרים, המתבצעים באותו פרק זמן, תופסים משאבים הדרושים לו.

07.06.39 **קִפָּאוֹן** **deadlock**

deadly embrace

מצב שבו עיבוד נתונים מעוכב מפני שכמה הֶתקנים או תהליכים בו-זמניים ממתינים למשאבים שהוקצו לתהליכים אחרים או בגלל תלות הדדית אחרת.

דוגמה:

מצב שבו תוכנית א' נועלת לשימוש בלבדי את רשומה 1, ומבקשת לנעול גם את רשומה 2 שהוקצתה לתוכנית ב'. באותו זמן תוכנית ב' נועלת לשימוש בלבדי את רשומה 2 ומחכה לאפשרות לנעול לשימושה הבלבדי את רשומה 1, בטרם תשחרר את רשומה 2.

07.06.40 **נְעִילָה** **lockout**

טכניקה להקצאת משאבים, שבה מגינים על משאבים משותפים על ידי כך שברגע מסוים רק הֶתקן אחד או תהליך אחד מקבלים הרשאת גישה אליהם.

דוגמה: מניעת קריאה של נתונים כל עוד מתבצע בהם עדכון.

07.06.41 **תָּכְנִית (תוכנית) אִתְחוּל** **bootstrap**
initial program load
IPL [abb]
תוכנית קצרה שהיא תושבת קבועה במחשב או נטענת בו בקלות ושביצועה מביא לזיכרון תוכנית גדולה יותר כגון מערכת הפעלה או מַטעֵן של מערכת הפעלה.

07.06.42 **אִתְחֵל** (1) **bootstrap** [v]
טען תוכנית אִתחוּל.

07.06.43 **מַטְעֵן אִתְחוּל** **bootsrap loader**
תוכנית קצרה המשמשת לטעינת תוכנית אִתחוּל.

07.06.44 **אִתְחֵל** (2) **boot** [v]
אִתחֵל מחשב על ידי טעינה מחדש של מערכת ההפעלה, וייתכן גם על ידי ניקוי זיכרון.

07.06.45 **חָרִיג** **exception**
מצב העלול להיווצר במהלך ביצוע תוכנית ולגרום סטייה מסדר הביצוע הרגיל, ואשר קיימים אמצעים להגדירו, להתריע על הופעתו, לזהותו, להתעלם ממנו או לטפל בו.
דוגמות:
תנאי (ON ERROR) בשפת PL/1; גלישת זיכרון; שגיאת טווח.

07.06.46 **הִתְרִיעַ עַל חָרִיג** **raise an exception (v)**
אותת על הופעתו של חריג בהתבסס על התרחשות מצב מסוים.

07.06.47 **מְטַפֵּל בַּחֲרִיגִים** **exception handler**
שִגרה המבוצעת בתגובה להתרחשות סוג מסוים של חריג.

07.06.48 **טִפֵּל בְּחָרִיג** **handle an exception (v)**
נקט פעולה ישירה כתוצאה מהתרחשות חריג.
הערה:
בדרך כלל הבקרה עוברת למטפל-חריגים, הנוקט פעולות דרושות.

07.06.49 **הֶעֱבִיר חָרִיג** **propagate an exception (v)**
בגלל חוסר יכולת טיפול בתוך מודול נתון, העביר את הבקרה למטפל בחריגים של המודול הקורא הקודם או של המודול המקונן, או התריע שוב במפורש על חריג בתוך מטפל-בחריגים אחר.

07.06.50 **חֲרִיג מְעִינָה** **addressing exception**
חריג המתרחש כשתוכנית מחשבת מען הנמצא מחוץ לגבולות מרחב המענים הזמינים שלה.

07.06.51 **חֲרִיג נְתוּנִים** **data exception**
חריג המתרחש כשתוכנית מנסה להשתמש בנתונים באופן שגוי או לגשת אליהם באופן שגוי.

07.06.52 **חֲרִיג פְּעוּלָה** **operation exception**
חריג המתרחש כשתוכנית מנסה לבצע פקודה שחלק-הפעולה שלה אינו חוקי.

07.06.53 **חֲרִיג הֲגָנָה** **protection exception**
חריג המתרחש כשתוכנית מנסה לגשת לאזור מוגן בהֶתקן אִחסון.

07.06.54 **חֲרִיג גְּלִישָׁה** **overflow exception**

חריג המתרחש כשתוצאה של פעולה גורמת גלישה.

07.06.55 **חֲרִיג חֲמִיקָה** **underflow exception**

חריג המתרחש כשתוצאה של פעולה גורמת חמיקה אריתמטית.

07.07 - נִפּוּי וּבְדוּק - Debugging and checking

07.07.01 **נִפָּה** **debug** [v]

גילה, איתר וסילק טעויות בתוכניות מחשב.

07.07.02 **תָּכְנִית (תוכנית) נִפּוּי** **debugger**

מְנַפֶּה

תוכנית-עזר לניפוי.

07.07.03 **הֵטִיל** **dump** [v]

רשם או הציג את התוכן של התקן אִחסון, כולו
או חלקו, ברגע מסוים, בתסדיר המקל על הניתוח.

דוגמה:

הטיל תוכן של התקני אחסון פנימיים, כגון זיכרון
ואוגרים כלליים, או תוכן של התקני אחסון
חיצוניים, כגון נתונים על דיסק או על סרט
מגנטי.

07.07.04 **הֶטְלָה** **dump (1)**

רישום או הצגה של התוכן של התקן אִחסון, כולו או חלקו, ברגע מסוים, בתסדיר המקל על הניתוח.

הערה:

הטלה תיעשה בדרך כלל למטרות ניפוי.

07.07.05 **הֶטֵל** **dump (2)**

הֶטֵל נְתוּנִים **data dump**

רשימה או תצוגה של נתונים המהווים את התוכן של התקן אִחסון, כולו או חלקו, ברגע מסוים, בתסדיר המקל על הניתוח.

07.07.06 **הֶטֵל בּוֹרֵר** **selective dump**

היטל של אתרי אִחסון מוגדרים בלבד.

07.07.07 **הֶטֵל שִׁנּוּיִים** **change dump**

היטל של אתרי אִחסון שתוכנם השתנה במשך זמן מוגדר.

07.07.08 **הֶטֵל סִיּוּם** **postmortem dump**

הֶטֵל סִיּוּם לֹא-תַּקִּין

היטל הנוצר בגלל סיום לא-תקין של ביצוע תוכנית.

07.07.09 **הֶטֵל בָּזָק** **snapshot dump**

העתק של הנתונים, כולם או חלקם, הנמצאים בזיכרון או במסד נתונים בנקודת זמן מסוימת.

07.07.10 **הֶטֵל זִכָּרוֹן** **memory dump**

היטל של תוכן הזיכרון הפנימי של מחשב.

הערה:

בדרך כלל בייצוג בינָרי, אוקטָלי או הקסָדסימָלי.

07.07.11 **בְּדוּק יְדָנִי** **desk checking**

טכניקת ניתוח סטטי, ייתכן גם הכוללת הדמיה ידנית, של ביצוע תוכנית שבה קוד מקור, תוצאות בדיקות או תיעוד אחר, נבחנים, בדרך כלל על ידי מי שייצר אותם, כדי לזהות תקלות, הפרות של תִקני פיתוח או בעיות אחרות.

07.07.12 **בִּצוּעַ מֻקְלָט** **playback**

טכניקה לרישום ההיסטוריה של ביצוע תוכנית או חלקה, באופן שניתן ליצור מחדש, תחת בקרה של המשתמש, את הקלט והפלט של התוכנית, לְפָנים או לאחור.

הערה:

בשימוש למטרות ניפוי.

07.07.13 **בִּצוּעַ חוֹזֵר** **replay**

טכניקת לכידה של קלט תוכנית, כך שניתן יהיה להזין בו מחדש תוכנית, ולבצע אותה בתנאים מבוקרים לשם ניתוח.

07.07.14 **פְּעֻלַּת צַעַד-צַעַד** **single-step operation**

בִּצוּעַ צַעַד-צַעַד **single-step execution**

step-by-step operation

אופן פעולה של מחשב, שבו בתגובה לאות חיצוני מתבצעת פקודה בודדת או חלק של פקודה.

הערה:

משמש לצורכי ניפוי.

07.07.15 **תָּכְנִית (תוכנית) אִבְחוּן** **diagnostic program**

תוכנית שנועדה לגלות, לאתר ולתאר תקלות בציוד או שגיאות בתוכניות אחרות.

07.07.16 **תָּכְנִית (תוכנית) מַעֲקָב** **trace program**

תוכנית המבצעת מעקב.

07.07.17 **מַלְכּוֹד קוֹד פְּעֻלָּה** **operation code trap**

שינוי מיוחד של חלק הפעולה של פקודת מכונה כך שבעת ביצוע הפקודה תיגרם הפסקת עבודה.

07.07.18 **תָּכְנִית (תוכנית) הַגָּהָה** **checking program**

תָּכְנִית (תוכנית) בְּדוּק

תוכנית אִבחון הבוחנת תוכניות מקור או נתונים בחיפוש אחר שגיאות תחביר וסמנטיקה או אי-עמידה בדרישות מוגדרות.

07.07.19 **טְלַאי** **patch**

שינוי ישיר של מודול יעד או של תוכנית טעונה, ללא איסוף או הידור חוזר של תוכנית מקור.

07.07.20 **הִטְלִיא** **patch** [v]

עשה שינוי ישיר של מודול יעד או של תוכנית טעונה, ללא איסוף או הידור חוזר של תוכנית מקור.

07.07.21 **טַעֲנַת תְּנַאי** **assertion**

קְבִיעָה

בתכנות מחשבים, מבנה לשוני המגדיר מצב מסוים החייב להתקיים או תנאי מסוים החייב להתממש בנקודה מסוימת בתוכנית, בעת הביצוע.

07.07.22 **טַעֲנַת לוּלָאָה** **loop assertion**

ראו 07.03.05.

07.07.23 **קָבוּעַ** [שם תואר] **invariant** [adj]

מִשְׁתַּמֵּר

התכונה של מה שמִשתמֵר בתוך סביבה מסוימת.

07.07.24 **קָבוּעַ לוּלָאָה** **loop invariant**

מצב או תנאי המשתמר במשך כל זמן ביצוע הלולאה.

07.07.25 **תְּנַאי-קֶדֶם** **precondition**

טענת תנאי המתייחסת לנקודה במעקובת (sequence) ביצוע התוכנית, הנמצאת מיד לפני חלק מוגדר של התוכנית.

07.07.26 **תְּנַאי-אַחַר** **postcondition**

טענת תנאי המתייחסת לנקודה במעקובת ביצוע התוכנית, הנמצאת מיד אחרי חלק מוגדר של התוכנית.

07.07.27 **הוֹכָחַת נְכוֹנוּת** (1) [הפעולה] **correctness proving**

פעולת ההוכחה המתמטית הפורמלית שהסמנטיקה של תוכנית עומדת בדרישות המפרט שלה.

07.07.28 **הוֹכָחַת נְכוֹנוּת** (2) [התוצאה] **proof of correctness**

התוצאה של פעולת ההוכחה המתמטית הפורמלית שהסמנטיקה של תוכנית עומדת בדרישות המפרט שלה.

07.07.29 **מִפְרָט פּוֹרְמָלִי** **formal specification**

מפרט כתוב בסימון פורמלי, המשמש לעתים קרובות להוכחת נכונות.

07.07.30 **נְכוֹנוּת חֶלְקִית** **partial correctness**

הוכחת נכונות (1), המציינת שטענות התנאי של פלט תוכנית נובעות לוגית מתוך טענות התנאי של הקלט שלה ומצעדי עיבוד.

07.07.31 **נְכוֹנוּת כּוֹלֶלֶת** **total correctness**

הוכחת נכונות (1), המציינת שטענות התנאי של פלט תוכנית נובעות לוגית מתוך טענות התנאי של הקלט שלה ומצעדי עיבוד, וכן שבעֵת סיומה, התוכנית עומדת בכל תנאֵי הקלט המוגדרים.

07.07.32 **זְרִיעַת תַּקָּלוֹת** **error seeding**

bug seeding

fault seeding

תהליך הכנסה מכוונת של תקלות ידועות, נוסף על תקלות שקיימות כבר בתוכנית, כדי לנטר את שיעור הגילוי והסילוק וכדי לאמוד את מספר התקלות שנותרו בה.

07.07.33 **תַּקָּלָה יְלִידָה** **indigenous error**

indigenous fault

תקלה בתוכנית, שלא הוכנסה במתכוון בתהליך זריעת תקלות.

07.07.34 **תָּכְנִית (תוכנית) בַּקָּרַת תַּקָּלוֹת** **error control software**

תוכנית לניטור מערכת מחשב כדי לגלות, לרשום ובמידת האפשר לתקן תקלות.

07.07.35 **תַּחֲזִית תַּקָּלוֹת** **error prediction**

טענה כמותית על אודות מספר התקלות הצפוי או סוגן, במערכת או ברכיב.

הערה:

הפעולה נקראת חיזוי תקלות.

07.07.36 **תַּקָּלָה לְלֹא תַּקָּנָה** **unrecoverable error**

תקלה שאי אפשר להתאושש ממנה ללא שימוש בטכניקות התאוששות חיצוניות לתוכנית המחשב.

07.08 - תִּכְנוּת מִיקְרוֹ - Microprogramming

07.08.01 **פְּקֻדַּת מִיקְרוֹ** **microinstruction**

הוראה המגדירה את הפעולה או הפעולות הבסיסיות הדרושות כדי לבצע פקודת מכונה או תִּפקוד חומרה עצמאי אחר, והמציינת את האופרנדים השייכים לפעולות אלה.

הערה:

פקודות מיקרו הן פקודות מכונה אמיתיות, וקוד מיקרו משמש ליצירת מכונה וירטואלית הנראית כבעלת אוצר פקודות יותר ידידותי למשתמש.

07.08.02 **תִּכְנוּת מִיקְרוֹ** **microprogramming**

תִּכנות המשתמש בפקודות מיקרו.

הערה:

תכנות מיקרו הוא תחליף לתיול (hard-wiring) של אותות הבקרה הדרושים לביצוע כל פקודת מכונה.

07.08.03 **תָּכְנִית (תוכנית) מִיקְרוֹ** **microprogram**

מעקובת פקודות מיקרו, שיחד עם רכיבי חומרה מתאימים, שולטת שליטה מלאה בביצוע של פקודת מכונה או בתִּפקוד חומרה עצמאי אחר.

07.08.04 **קוֹד מִיקְרוֹ** **microcode**

אוסף פקודות מיקרו המהווה חלק של תוכנית מיקרו, תוכנית מיקרו שלמה או קבוצת תוכניות מיקרו.

07.08.05 **מְאַסֵּף קוֹד-מִיקְרוֹ** **microcode assembler**

תוכנית המתרגמת תוכניות מיקרו מצורה סִמלית לצורה בינָרית.

07.08.06 **פְּעֻלַּת מִיקְרוֹ** **microoperation**

בתִּכנות מיקרו, אחת הפעולות הבסיסיות הדרושות לביצוע פקודת מכונה או לביצוע תִּפקוד חומרה עצמאי אחר.

07.08.07 **מַחְשֵׁב מְבֻקַּר-תָּכְנִית-מִיקְרוֹ** **microprogrammable computer**

מחשב שהמשתמש יכול ליצור בו תוכניות מיקרו או לשנותן.

07.09 - פְּקֻדּוֹת וּמְעָנִים - Instructions and addresses

07.09.01 **פְּקֻדָּה** **instruction**

הוֹרָאָה

פירוט של פעולה וזיהוי האופרנדים המסומכים אליה.

07.09.02 **פְּקֻדַּת מְכוֹנָה** **machine instruction**

פקודה שניתן לבצעה ישירות על ידי המחשב, ללא תרגום.

הערה:

פקודת מכונה היא חלק משפת מכונה.

07.09.03 **תַּסְדִּיר פְּקֻדָּה** **instruction format**

מערך החלקים המרכיבים פקודה.

07.09.04 **אוֹצַר פְּקֻדּוֹת** **instruction set**

instruction repertoire

קבוצת כל הפקודות המוכרות על ידי מחשב נתון או הכלולות בשפת תִּכנות נתונה.

07.09.05 **אֹרֶךְ פְּקֻדָּה** **instruction lentgh**

מספר המילים, הבתים או הסיביות, הדרוש לאִחסון פקודת מכונה.

07.09.06 **חֵלֶק פְּעֻלָּה** **operation part**

operation field

החלק של פקודת מכונה או של פקודת מיקרו, המפרֵט את הפעולה שיש לבצע.

07.09.07 **מַעַן** **address**

ערך המזהה מקום.

הערה:

מקובל בציבור גם "כתובת".

דוגמות:

מספר אוגר, המען של חלק מסוים של התקן אחסון, המען של הֶתקן, מען ברשת תקשורת מחשבים.

07.09.08 **חֵלֶק מַעַן** **address part**

החלק של פקודת מכונה או של פקודת מיקרו המפרֵט את המען של אופרנד.

07.09.09 **תַּסְדִּיר מַעַן** **address format**

המספר והסידור של האלמנטים המרכיבים מען.

דוגמות:

1. הדף וההיסט (offset) במערכת מעינה וירטואלית;
2. המען של ערוץ, הֶתקן, רשומה.

07.09.10 **קוֹד פְּקֻדָּה** **instruction code**

קבוצת בתים ליצוג פקודות המכונה השונות המותרות של מחשב מסוים.

07.09.11 **קוֹד פְּעֻלָּה** **operation code**

הייצוג המקודד של חלק הפעולה של פקודת מכונה.

דוגמה: ברמת שפת-סף, BNZ משמשת לציון הפעולה BRANCH IF NOT ZERO המיוצגת בסופו של דבר ברמת קוד מכונה כדגֵם סיביות מוגדר.

07.09.12 **פְּקֻדָּה לְלֹא מַעַן** **zero-address instruction**

פקודה שאין בה חלק-מען.

דוגמה:

פקודת HALT.

07.09.13 **פְּקֻדַּת מַעַן יָחִיד** **one-address instruction**
single-address instruction
single-operand instruction

פקודה הכוללת חלק-מען אחד.

דוגמה:

פקודה לטעון באוגר את התוכן של מקום A בזיכרון.

07.09.14 **פְּקֻדַּת שְׁנֵי מְעָנִים** **two-address instruction**
double-operand instruction

פקודה הכוללת שני מענים, שלכל אחד מהם חלק-מען משלו.

דוגמה:

הפקודה להוסיף את התוכן של מקום A בזיכרון לתוכן של מקום B בזיכרון.

07.09.15 **פְּקֻדַּת שְׁלֹשָׁה מְעָנִים** **three-address instruction**

פקודה הכוללת שלושה מענים, שלכל אחד מהם חלק-מען משלו.

דוגמה:

הפקודה להוסיף את התוכן של מקום A בזיכרון לתוכן של מקום B בזיכרון ולאחסן את התוצאה במקום C בזיכרון.

07.09.16 **פְּקֻדָּה רַבַּת-מְעָנִים** **n-address instruction**

פקודה הכוללת מספר רב של מענים, שלכל אחד מהם חלק-מען משלו.

07.09.17 **פְּקֻדַּת מַעַן אֶחָד-וְעוֹד-אֶחָד** **one-plus-one address instruction**

פקודה הכוללת שני חלקי-מען, שבה חלק המען השני כולל את מען הפקודה הבאה לביצוע.

דוגמה:

הפקודה לטעון באוגר את התוכן של מקום A בזיכרון ואחר כך לבצע את הפקודה שבמקום B בזיכרון.

07.09.18 **מְעִינָה מִשְׁתַּמֵּעַת** **implicit addressing**
implied addressing

שיטת-מעינה שבה המען של האופרנד משתמע מחלק-הפעולה של הפקודה.

דוגמה:

במחשב בעל אוגר יחיד, פקודה המתייחסת לאוגר אינה זקוקה למען.

07.09.19 **מְעִינָה עוֹקֶבֶת** **one-ahead addressing**

שיטת מעינה משתמעת, המניחה שהאופרנדים של הפקודה הנוכחית נמצאים בזיכרון בהמשך למקום האופרנדים ששימשו בביצוע הפקודה הקודמת.

07.09.20 **מְעִינָה נִשְׁנִית** **repetitive addressing**

שיטת מעינה משתמעת, המניחה שהפעולה של הפקודה הנוכחית מתייחסת לאופרנדים של הפקודה הקודמת שבוצעה.

07.09.21 **פְּקֻדָּה יְשִׁירָה** **direct instruction**

פְּקֻדָּה מִיָּדִית **immediate instruction**

פקודה הכוללת את ערך האופרנד, ולא את מענו.

07.09.22 **אוֹפֶרַנְד מִיָּדִי** **immediate operand**

אופרנד שבפקודה נמצא ערכו ולא מענו.

07.09.23 **נְתוּן מִיָּדִי** **immediate data**

נתון הנמצא בפקודה.

07.09.24 **פְּקֻדָּה עֲקִיפָה** **indirect instruction**

פקודה הכוללת מען עקיף.

07.09.25 **פְּקֻדַּת סְרָק** **no-op**

no-operation instruction

פקודה שביצועה גורם למחשב רק להתקדם לפקודה הבאה.

07.09.26 **פְּקֻדָּה מְיֻחֶסֶת** **privileged instruction**

פקודה הניתנת לביצוע רק באופן-תפעול (operation mode) מוגדר.

07.09.27 **פְּקֻדַּת קְפִיצָה** **jump instruction**

פקודה המציינת קפיצה.

07.09.28 **פְּקֻדַּת קְפִיצָה לֹא-מֻתְנֵית** **unconditional jump instruction**

פקודת-קפיצה המציינת קפיצת חובה.

07.09.29 **פְּקֻדַּת קְפִיצָה מֻתְנֵית** **conditional jump instruction**

פקודת-קפיצה המגדירה תנאי לביצוע הקפיצה.

07.09.30 **מַעֲקֹבֶת קְרִיאָה** **calling sequence**

סִדרת פקודות המביאה לביצוע של שגרה, המספקת לה במידת הצורך נתונים לעיבוד והמבקרת את משלוח התוצאות, אם ישנן, ואת החֲזרה לתוכנית הקוראת.

07.09.31 **מֶרְחַב מְעָנִים** **address space**

מֶרְחַב מְעִינָה

קבוצת המענים העומדים לרשות תוכנית מסוימת או לרשות יחידה תִפקודית.

הערה:

מרחב מענים עשוי לכלול מענים וירטואליים.

07.09.32 **מַעַן סִמְלִי** **symbolic address**

מזהה המייצג מען.

07.09.33 **מַעַן יָשִׁיר** **direct address**

מען המזהה ישירות מקום בזיכרון (בשונה ממען עקיף).

הערה:

המקום עשוי להיות מקום בַּזיכרון או התקן.

07.09.34 **מַעַן-בָּסִיס** **base address**

מען המשמש נקודת מוצא לחישוב מענים.

07.09.35 **מַעַן מֻחְלָט** **absolute address**

מען ישיר המזהה מקום בלי להתייחס למען-בסיס.

הערה:

מען מוחלט עשוי לשמש כמען-בסיס.

07.09.36 **מַעַן יַחֲסִי** **relative address**

מען ישיר המזהה מקום לפי מרחקו ממען הבסיס.

07.09.37 **מַעַן עָקִיף** **indirect address**

multilevel address

מען המזהה מקום של מען אחר.

הערה:

המקום המזוהה עשוי לכלול את המען של האופרנד הנדרש או מען עקיף נוסף; שרשרת מענים זו תוביל בסופו של דבר לַאופרנד.

07.09.38 **מַעַן זָחִיחַ** **relocatable address**

מען שיש לעדכנו כשמזיחים את הנתונים שהוא מתייחס אליהם או כשמזיחים את התוכנית המכילה אותו.

07.09.39 **מַעַן מְחֻשָּׁב** **generated address**

מען שחושב במהלך ביצוע תוכנית.

07.09.40 **מַעַן בְּפֹעַל** **effective address**

המען המתקבל באמצעות מִפתוּח, מעינה עקיפה או שינוי אחר של מען מסוים.

הערה:

אם המען המסוים אינו דורש שינוי כלשהו, הריהו גם המען בפועל.

07.09.41 **הַסְגָּלַת מַעַן** **address modification**

כל פעולה אריתמטית, לוגית או תחבירית המבוצעת במען.

07.09.42 **מַעַן וִירְטוּאָלִי** **virtual address**

במערכת זיכרון וירטואלי, המען המוקצה למָקום בהחסן חיצוני כדי לאפשר גישה אליו כאילו היה חלק מהזיכרון הראשי.

07.09.43 **מַעַן מַמָּשִׁי** **real address**

המען של מקום בחלק הזיכרון הראשי של מערכת זיכרון וירטואלי.

07.09.44 **אִינְדֶּקְס** **index** [in computer programming]

צִיּוּן

בתכנות מחשבים, מספר שלם המזהה את מקומו של פריט נתונים במעקובת של פריטי נתונים.

07.09.45 **מַעַן מְמֻפְתָּח** **indexed address**

מען שמוסיפים לו את התוכן של אוגר-אינדקס אחד או יותר.

07.09.46 **מַעַן יַחֲסִי-לְעַצְמוֹ** **self-relative address**

מען שמוסיפים אותו למען הפקודה שבה הוא מופיע, כדי לקבל את מען המקום בזיכרון, שאליו יש לגשת.

07.09.47 **תַּרְשִׁים מִבְנֶה** **structure chart**

תַּרְשִׁים מִדְרָג **hierarchy chart**

תרשים המזהה מודולים, פעילויות או ישויות אחרות במערכת או בתוכנית, והמראה כיצד ישויות גדולות או כלליות יותר מתפרקות ליישויות קטנות ומפורטות יותר.

הערות:

1. תרשים מבנה אינו זהה בהכרח לתרשים קריאה (call graph).
2. ראו ציור 1.

07.09.48 **תַּרְשִׁים קְרִיאָה** **call graph**

עֵץ קְרִיאָה **call tree**

תרשים המזהה מודולים במערכת או בתוכנית ומציג איזה מודולים קוראים זה לזה.

הערות:

1. תרשים קריאה אינו זהה בהכרח לתרשים מבנה.
2. ראו ציור 2.

07.09.49 **תַּרְשִׁים זְרִימַת בַּקָּרָה** **control flow diagram**

control flow graph

תרשים המתאר את קבוצת כל המעקובות האפשריות שניתן לבצע בהן פעולות במהלך ביצוע תוכנית.

07.09.50 **תַּרְשִׁים תֵּבוֹת** **box diagram**

Chapin chart

Nassi-Shneiderman chart

תרשים זרימת בקרה המורכב מתיבות עוקבות ומקוננות המייצגות צעדים עוקבים, חזרות ופסוקי תנאי.

הערה:

ראו ציור 3.

07.09.51 **תַּרְשִׁים זְרִימַת נְתוּנִים** **data flow diagram**

data flowchart

data flow graph

תרשים שבו צמתים מתארים מקורות נתונים, קולטי נתונים, החסני נתונים ותהליכי עיבוד נתונים, ושבו קווי קישור בין הצמתים מתארים זרימת נתונים לוגית.

הערה:

ראו ציור 4.

07.09.52 **תַּרְשִׁים בּוּעוֹת** **bubble chart**

תרשים שבו ישויות מתוארות על ידי מעגלים (בועות), והיחסות (relationships) ביניהן מיוצגות על ידי קווי קישור המשוכים בין המעגלים.

הערה:

ראו ציור 5.

07.09.53 **תַּרְשִׁים קֶלֶט-עִבּוּד-פֶּלֶט** **input-process-output chart**

IPO chart

תרשים של מערכת תוכנה או של מודול, המורכב מחלקים אלה: מלבן בצד שמאל, המכיל רשימת קלטים, מלבן במרכז המכיל רשימת צעדי עיבוד, מלבן בצד ימין המכיל רשימת פלטים וחיצים המחברים קלטים לצעדי עיבוד וצעדי עיבוד לפלטים.

הערה:

ראו ציור 6.

07.09.54 **תַּרְשִׁים הֶחְלֵף מַצָּבִים** **state transition diagram**

תַּרְשִׁים מַצָּבִים **state diagram**

תרשים המתאר את המצבים שמערכת או רכיב יכולים להימצא בהם, ומראה את האירועים או הנסיבות שגרמו למעבר ממצב אחד למצב אחר או את התוצאות של מעבר כזה.

07.10 - תַּהֲלִיכִים בּוֹ-זְמַנִּיִּים - Concurrent processes

07.10.01 **מַצַּב מַטָּלָה** **task state**

מַצַּב מְשִׂימָה

אחד המצבים שמשימה עשויה להימצא בהם במשך חייה.

07.10.02 **הַפְעָלָה** **activation** [in computer programming]

בתִכנוּת מחשבים, יצירת רשומת הפעלה.

הערה:

במונחי בינה מלאכותית נקבע "שפעול".

07.10.03 **שִׁפְרוּט** **elaboration**

התהליך שבו הכרזה מביאה, עוד לפני תחילת הביצוע של תוכנית, לידי תוצא כגון יישוב התייחסויות, בידוק סוגי נתונים או הקצאת זיכרון.

הערה:

ראו ציור 7.

07.10.04 **בָּצִיעַ** **executable**

מתייחס למצב-משימה שבו משימה הופעלה וטרם הושלמה.

הערות:

1. משימה בציעה יכולה להיות: מוכנה, בריצה או חסומה.
2. ראו ציור 7.
3. בשפת **ADA** משמש המונח "קריא" (callable).

07.10.05 **חָסוּם** **blocked**

מתייחס למצב-משימה שבו משימה בציעה מושהית או ממתינה לאירוע.

הערה:

ראו ציור 7.

07.10.06 **מוּכָן** **ready**

מתייחס למצב-משימה שבו משימה בציעה ממתינה להרצה ואינה חסומה.

הערה:

ראו ציור 7.

07.10.07 **רָץ** **running**

מתייחס למצב-משימה שבו משימה בציעה מתבצעת כעת.

הערה:

ראו ציור 7.

07.10.08 **מֻשְׁהֶה** **delayed**

מתייחס למשימה בציעה שנחסמה על ידי הוראת השהיה.

הערה:

ראו ציור 7.

07.10.09 **נִשְׁלַם** **completed**

מתייחס למצב-משימה שבו ביצוע המשימה נגמר וכל האירועים התלויים בה נפתרו.

הערות:

1. בשפת **ADA** התרעה על חריג במהלך הפעלה יהיה אירוע נדיר שיגרום להשלמת משימה.
2. ראו ציור 7.

07.10.10 **גָּמוּר** **terminated**

מתייחס למצב-משימה שבו ביצוע משימה נשלם, כל האירועים התלויים בה נפתרו ורשומת ההפעלה שוחררה.

הערה:

ראו ציור 7.

07.10.11 **מְשִׁימַת אָב** **master task**

מַטְלַת אָב

מודול של תוכנית שביצועו יוצר משימה.

07.10.12 **כְּנִיסַת מְשִׁימָה** **task entry**

מקום במשימה המספק מִנשָׂק למודול קורא.

07.10.13 **שׁוֹעֵר** **guard**

בפקודת-המתנה בוררת, ביטוי תנאי המשמש לקביעה אם הכניסה פתוחה או סגורה.

07.10.14 **שׁוֹעֵר פָּתוּחַ** **open guard**

שוער שתוצאת התנאי שלו "אמת".

07.10.15 **שׁוֹעֵר סָגוּר** **closed guard**

שוער שתוצאת התנאי שלו "שקר".

07.10.16 **תַּהֲלִיכוֹן** **thread**

תהליך (2) בתוך תהליך אחר, המשתמש לצרכיו במשאבים של התהליך האחר.

07.11 - סְבִיבוֹת תְּמִיכָה - Support environments

07.11.01 **רְכִיב חֲלוּפִי** **stub**

רכיב בתוכנית, המשמש זמנית לקידום פיתוחה.

דוגמה:

בהידור או בבדיקה משתמשים ברכיב חלופי עד שהרכיב הממשי נעשה זמין.

07.11.02 **פִּגּוּם** **scaffolding**

תוכניות ונתונים שנתכנו כדי לתמוך בפיתוח ובבדיקה של תוכנה, אבל אינם אמורים להיות חלק מהמוצר הסופי.

דוגמות:

שגרות דמה או קובצי דמה, מחוללים של אירועי מבחן, מוניטורים של תוכנה, רכיבים תחליפיים.

07.11.03 **מַעֲרֶכֶת תִּכְנוּת** **programming system**

בסביבת תִּכנות, שפות התִּכנות וכלֵי התוכנה הדרושים לפיתוח תוכניות הכתובות בשפות התכנות ולשימוש בַּתוכניות.

07.11.04 **מַזְלֵג** **fork**

שכפול והפעלה של תהליך על ידי התהליך עצמו וסביבת הביצוע שלו.

07.11.05 **סִפְרִיַּת תָּכְנִיּוֹת** **program library**

אוסף מאורגן של תוכניות או של חלקי תוכניות, וייתכן גם של מידע המתייחס לשימוש בהם.

הערה:

ספריית תוכניות מכונה בדרך-כלל לפי התוכניות המאוחסנות בה, כגון: ספריית תוכניות מקור, ספריית נהלים.

07.11.06 **סִפְרִיַּת תָּכְנָה** **software library**

אוסף מבוקר של תוכנה והתיעוד המתייחס אליה, המיועד לסייע בפיתוח תוכנה, בשימוש בה ובתִּחזוקה.

07.11.07 **סִפְרִיַּת מַעֲרֶכֶת** **system library**

ספריית תוכנה שהיא תושבת בתוכנה של מערכת ענ"א ושניתן לגשת אליה לצורך שימוש בה או לשלב אותה בתוכניות אחרות על ידי התייחסות.

דוגמה:

ספריית מַקרו.

07.11.08 **סְבִיבַת תִּפְעוּל** **operating environment**

הסביבה החיצונית של תוכנית, המתקיימת או האמורה להתקיים במהלך ביצוע התוכנית.

07.11.09 **סְבִיבַת עִבּוּד-אֲצָוּוֹת** **batch processing environment**

סביבת תִּפעוּל שבה נתוני קלט אינם מעובדים בעת הגעתם אלא נאספים ומעובדים במנות.

07.11.10 **סְבִיבָה הִדּוּדִית** **interactive environment**

סביבת תִּפעוּל שבה תוכנית במהלך הביצוע שלה מגיבה לכל משתמש, ושבה המשתמש חש שהוא משפיע ישירות על הפעולות במהלך העיבוד.

07.11.11 **סְבִיבַת זְמַן-אֱמֶת** **real-time environment**

סביבת תִּפעוּל התומכת בביצוע תוכניות זמן-אמת.

07.11.12 **תָּכְנִית (תּוֹכנית) שֵׁרוּת** **utility program**

תוכנית המספקת שירותים כלליים הנדרשים לעִתים קרובות למשתמשי מחשב ולצוות תחזוקה.

דוגמות:

תוכנית אבחון, תוכנית מעקב, תוכנית מיון.

07.11.13 **שִׁגְרַת שֵׁרוּת** **utility routine**

שִׁגרה המספקת שירותים כלליים הנדרשים לעִתים קרובות למשתמשי מחשב ולצוות תחזוקה.

דוגמה:

שגרת קלט.

07.12 - מַטָּרוֹת וְעֶקְרוֹנוֹת - Goals and principles

07.12.01 **מֵסְגָּלוּת** **modifiability**

מידת הקלות לשנות תוכנית.

07.12.02 **יְעִילוּת** **efficiency**

מידת מהירות הביצוע של תוכנית או מידת צריכת הזיכרון שלה, או שתיהן.

07.12.03 **מוּבָנוּת** **understandability**

מידת הקלות שבה יכול אדם לקרוא תוכנית ולזהות מתוך הפתרון את מיפוי מבני הנתונים, עצמי הנתונים והאלגוריתמים, אל העולם הממשי.

07.12.04 **מוֹדוּלָרִיוּת** **modularity**

המידה שבה תוכנית מורכבת ממודולים, כך ששינוי במודול אחד משפיע השפעה מִזערית על מודולים אחרים.

07.12.05 **לְכִידוּת** **cohesion**

קוֹהֶזְיָה **module strength**

האופן והשיעור שבהם פעילויות של מודול אחד מתייחסות זו לזו.

הערות:

1. לכידות חזקה משמעותה יחסים הדוקים מאוד בין הפעילויות במודול.
2. ניתן לדרג סוגי לכידות מן החזָקה אל החלשה כלהלן:
   - לכידות תִפקודית
   - לכידות מידעית
   - לכידות תִקשורתית
   - לכידות בזמן
   - לכידות לוגית
   - לכידות מִקרית
3. בשונה מ"**צימוד**"

07.12.06 **לְכִידוּת תִּפְקוּדִית** **functional cohesion**

לכידות שבה כל הפעילויות של מודול תורמות לביצוע מטרה יחידה מוגדרת.

07.12.07 **לְכִידוּת מֵידָעִית** **informational cohesion**

לכידות שבה כל הפעילויות של מודול מבוצעות על גבי מבנה נתונים משותף, אך כל אחת מהן משתמשת בנקודת כניסה לא תלויה ובקוד לא תלוי.

07.12.08 **לְכִידוּת תִּקְשָׁרְתִּית (תקשורתית)** **communicational cohesion**

לכידות שבה הפעילויות של מודול משתמשות באותם נתוני קלט או תורמות ליצירת אותם נתוני פלט.

07.12.09 **לְכִידוּת בַּזְּמַן** **temporal cohesion**

לכידות שבה כל הפעילויות של מודול דרושות בזמן נתון.

דוגמה:

מודול הכולל את כל פעילויות האִתחול של תוכנית.

07.12.10 **לְכִידוּת לוֹגִית** **logical cohesion**

לכידות שבה כל הפעילויות של מודול דומות לוגית.

דוגמה:

עיבוד של נתונים ממצעי נתונים שונים, במודול אחד.

07.12.11 **לְכִידוּת מִקְרִית** **coincidental cohesion**

לכידות שבה אין יוחסה (relationship) תפקודית הדדית בין הפעילויות של מודול.

07.12.12 **לְכִידוּת נֶהֱלִית (נוֹהוֹלִית)** **procedural cohesion**

לכידות שבה כל הפעילויות של מודול תורמות לנוהל נתון, כגון חִזרור או החלטה.

07.12.13 **לְכִידוּת עוֹקֶבֶת** **sequential cohesion**

לכידות שבה התוצאה של פעילות אחת במודול משמשת אופרנד לַפעילות העוקבת באותו מודול.

07.12.14 **צִמוּד** **coupling**

חִברוּר או תלות הדדית בין מודולים שונים.

הערות:

1. צימוד רופף פירושו היעדר תלות הדדית או חִברוּר בין מודולים או קיומם של תלות או חִברוּר חלשים.
2. ניתן לדרג סוגי צימוד מן הרופֵף עד ההדוק כלהלן:
   - בלי צימוד
   - צימוד נתונים
   - צימוד בקרה
   - צימוד חיצוני
   - צימוד משותף
   - צימוד תוכן
3. בשונה מ"לכידות" (07.12.05).

07.12.15 **צִמוּד נְתוּנִים** **data coupling**

צימוד שבו הנתונים מתחלקים בין מודולים.

07.12.16 **צִמוּד בַּקָּרָה** **control coupling**

צימוד שבו מודול אחד מעביר נתונים למודול אחר במטרה מפורשת להשפיע על פעולת המודול האחר.

07.12.17 **צִמוּד חִיצוֹנִי** **external coupling**

צימוד שבו ניתן לשלוט בצימוד של משתנים על ידי הגבלתו למשתנים שהוכרזו פורמלית כחיצוניים.

הערה:

PL/1 היא אחת משפות התכנות בעלות יכולת זאת.

07.12.18 **צִמוּד מְשֻׁתָּף** **common-environment coupling**
common coupling

צימוד שבו מודולים ניגשים לנתונים משותפים.

07.12.19 **צִמוּד תֹּכֶן** **content coupling**

צימוד שבו מודול אחד מתייחס למודול אחר או משנה את הקוד של מודול אחר.

07.12.20 **מְנִיפָה מִתְכַּנֶּסֶת** **fan-in**

מספר המודולים השולטים במודול נתון.

הערה:

ערך גבוה של מניפה מתכנסת מצביע על צימוד הדוק, משום שהוא מידה של התלות בין המודולים.

07.12.21 **מְנִיפָה מִתְבַּדֶּרֶת** **fan-out**

מספר המודולים הנשלטים על ידי מודול נתון.

הערה:

ערך גבוה של מניפה מתבדרת מצביע על רמת מורכבות גבוהה של המודול הקורא, וזאת בגלל המורכבות של הלוגיקה הדרושה לבקרה של רכיבים כפיפים ולתיאום ביניהם.

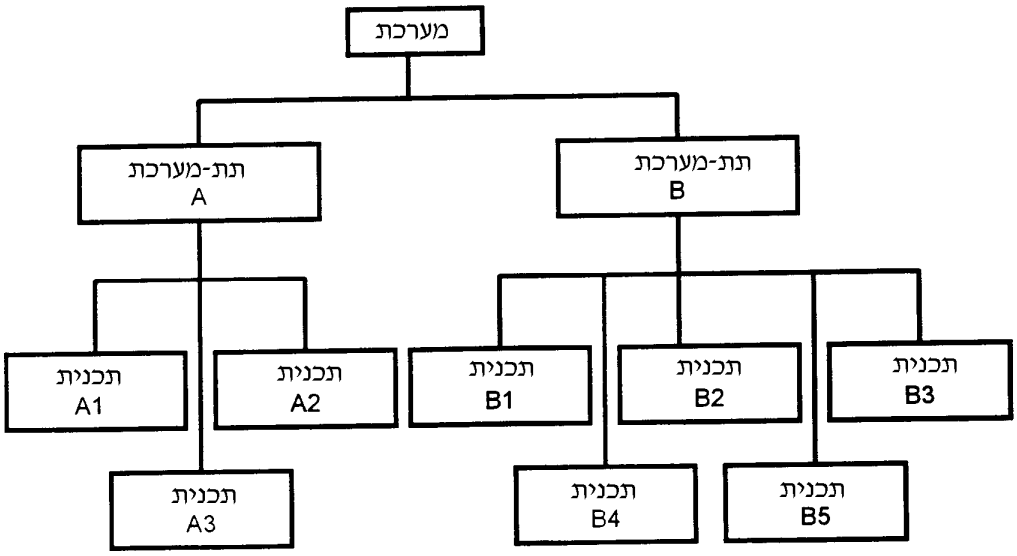
07.12.22 **אִתּוּר** **localization**

העיקרון המיושם בקבוצה של מודולים שהם בעלי לכידות חזקה וצימוד רופף.

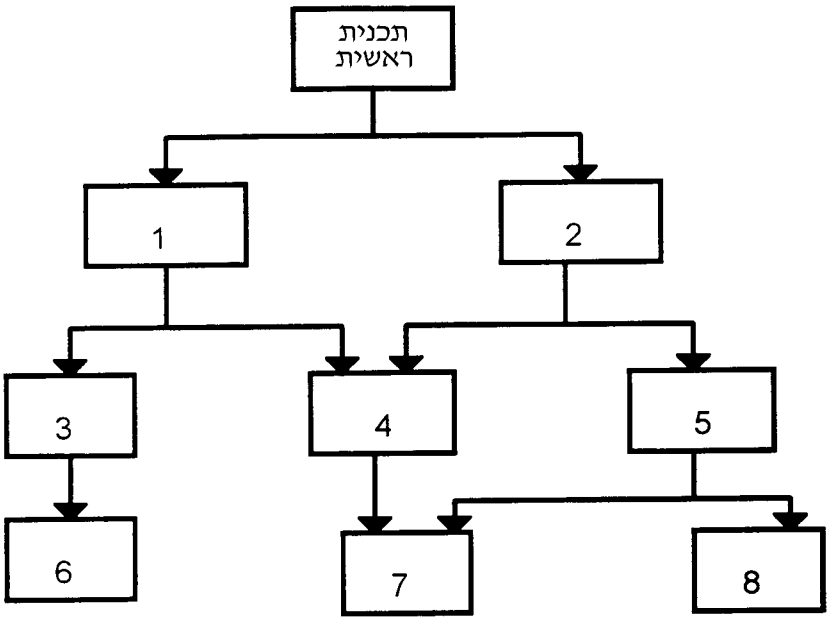
07.12.23 **מַדַּד אִשּׁוּר** **confirmability**

המידה שבה התֶכֶן והמבנה של תוכנית נעשו כך שניתן לבדוק את כל חלקיה ללא קושי.

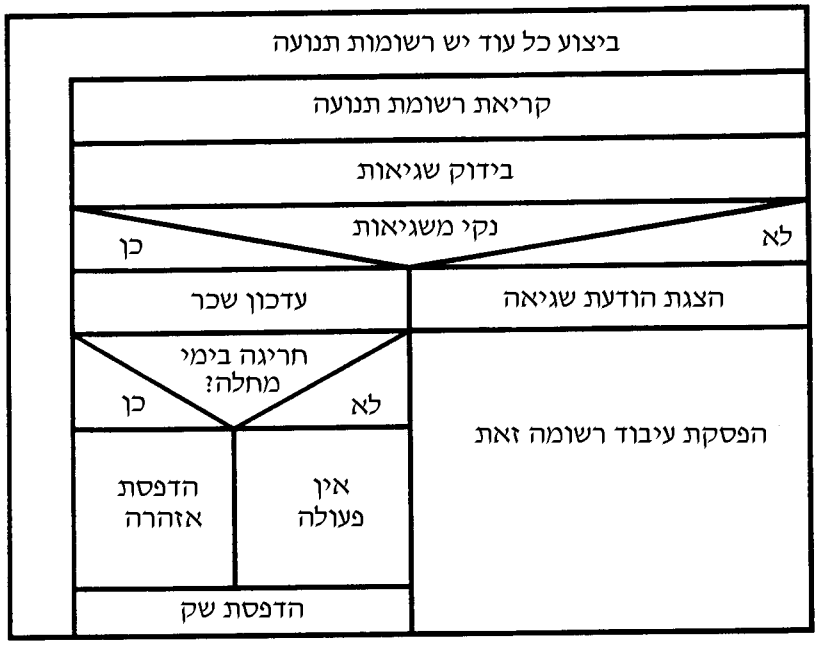
איורים



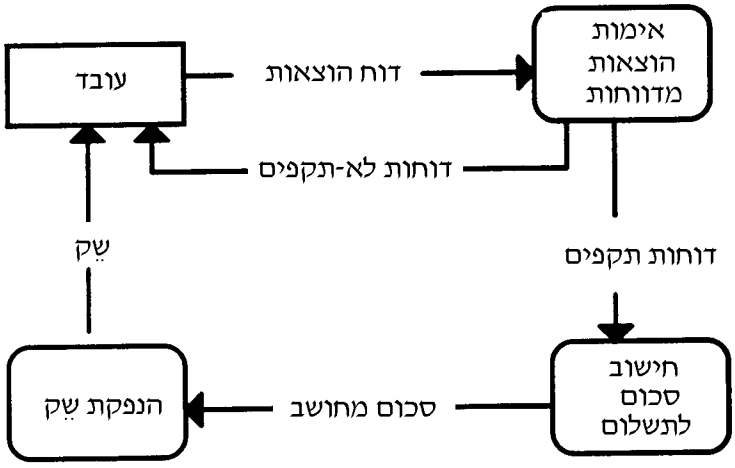
ציור 1 - **תרשים מבנה**



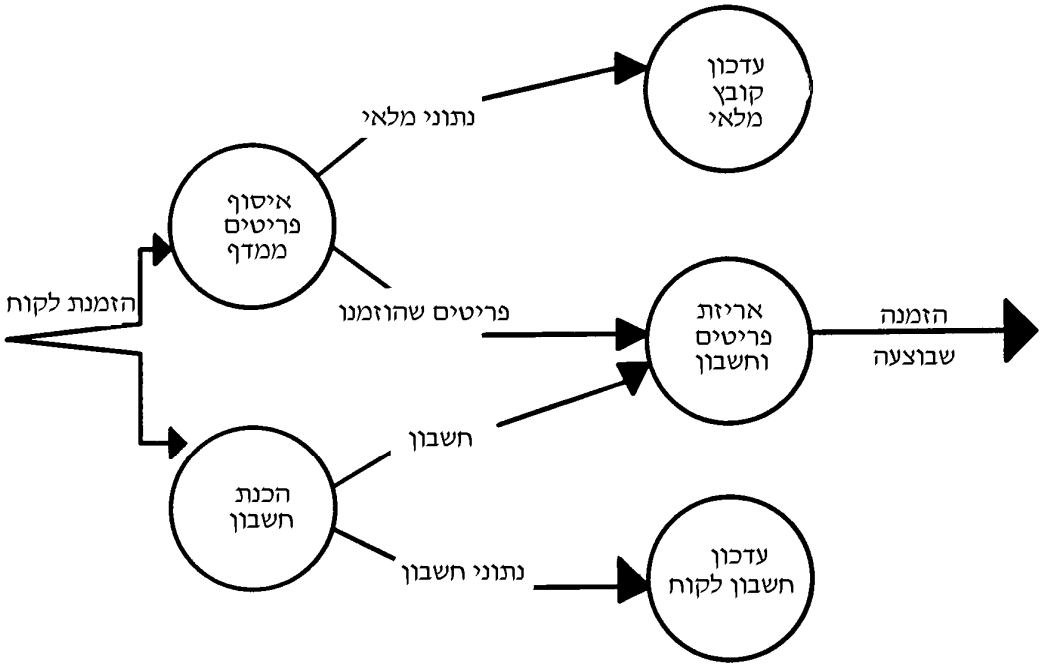
ציור 2 - **תרשים קריאה**



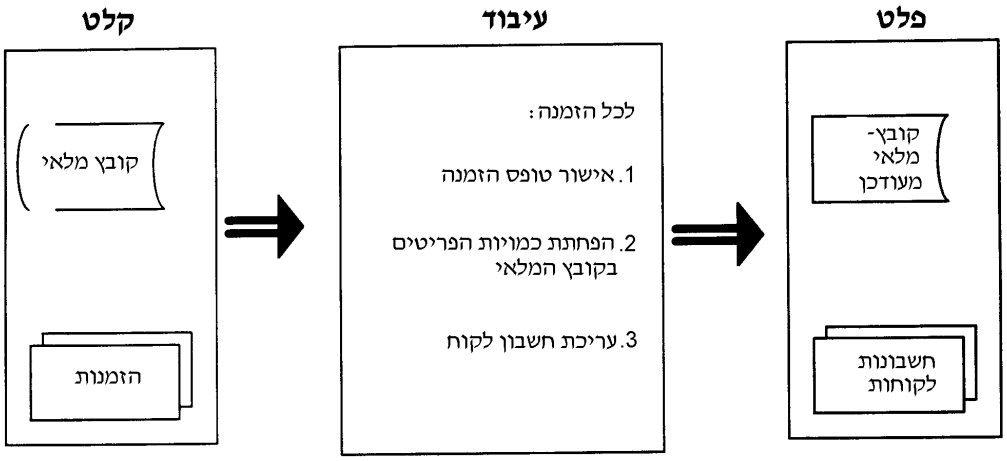
ציור 3 - **תרשים תיבות**



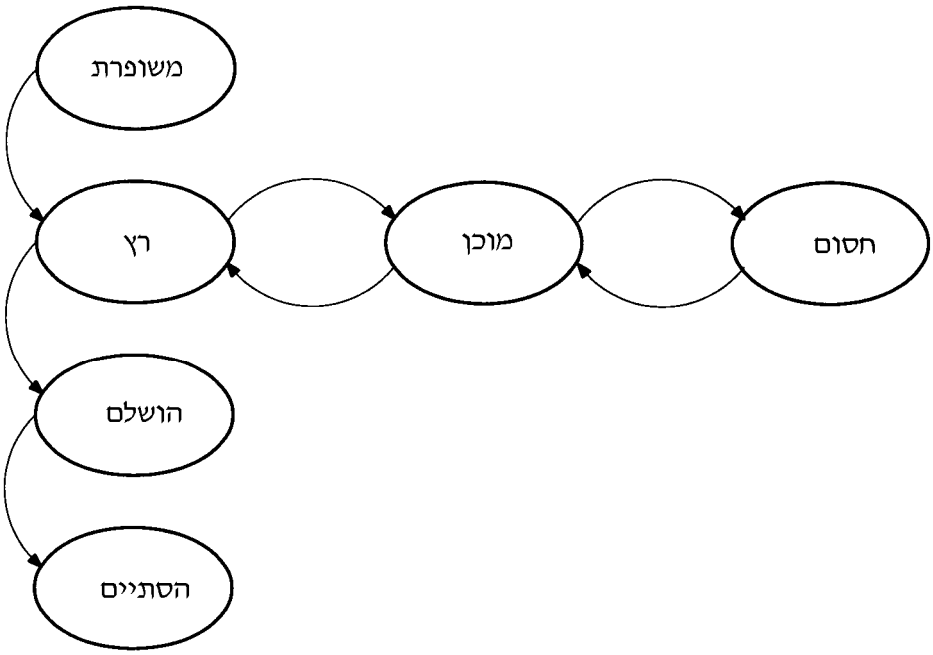
ציור 4 - **תרשים זרימת נתונים**



ציור 5 - תרשים בועות



ציור 6 - תרשים קלט-עיבוד-פלט



ציור 7 - **תרשים שינוי מצבי-משימות**

מפתח מונחים א-ב

English Term	מונח עברי	מספר מונח
immediate operand	אוֹפֶרַנְד מִיָדִי	07.09.22
instruction set, instruction repertoire	אוֹצַר פְּקֻדּוֹת	07.09.04
iteration	אִיטֶרַצְיָה	07.03.01
index [in computer programming]	אִינְדֶקֶס	07.09.44
assemble [v]	אִסֵף	07 04.08
assemble-and-go	אַסֵף וּבַצַע	07.04.18
instruction lentgh	אוֹרֶךְ פְּקֻדָּה	07.09.05
localization	אִתּוּר	07.12.22
bootstrap [v]	אִתְחֵל (1)	07.06.42
boot [v]	אִתְחֵל (2)	07.06.44
desk checking	בְּדוּק יְדָנִי	07.07.11
loaded origin	בְּסִיס טְעִינָה	07.05.10
assembled origin	בְּסִיס-סַף	07.04.15
replay	בְּצוּעַ חוֹזֵר	07.07.13
playback	בְּצוּעַ מֻקְלָט	07.07.12
symbolic execution	בְּצוּעַ סִמְלִי	07.06.13
single-step execution	בְּצוּעַ צַעַד-צַעַד	07.07.14
executable	בָּצִיעַ	07.10.04
loop control	בַּקָּרַת לוּלָאָה	07.03.07
rehashing	גִּבּוּב חוֹזֵר	07.02.22
hashing, hash addressing	גִּבּוּב, מְעִינָה מְגֻבָּבָת	07.02.18
loop body	גּוּף הַלּוּלָאָה	07.03.06
terminated	גָּמוּר	07.10.10
flag	דֶּגֶל	07.02.13
compilation	הִדּוּר	07.04.21
independent compilation	הִדּוּר לֹא-תָּלוּי	07.04.71
dependent compilation	הִדּוּר תָּלוּי	07.04.70
separate compilation	הִדּוּר תָּלוּי	07.04.70
compile [v]	הִדֵּר	07.04.19
compile-and-go	הַדֵּר וּבַצַע	07.04.26
correctness proving	הוֹכָחַת נְכוֹנוּת (1) [הפעולה]	07.07.27
proof of correctness	הוֹכָחַת נְכוֹנוּת (2) [התוצאה]	07.07.28
instruction	הוֹרָאָה	07.09.01
relocate [v]	הִזִּיחַ	07.05.12
dump [v]	הֵטִיל	07.07.03
selective dump	הֶטֵל בּוֹרֵר	07.07.06
snapshot dump	הֶטֵל בָּזָק	07.07.09
memory dump	הֶטֵל זִכָּרוֹן	07.07.10

English Term	מונח עברי	מספר מונח
postmortem dump	הֶטֵל סִיּוּם, הֶטֵל סִיּוּם לֹא-תַּקִּין	07.07.08
change dump	הֶטֵל שִׁנּוּיִים	07.07.07
dump (2), data dump	הֶטֵל, הֶטֵל נְתוּנִים	07.07.05
dump (1)	הֶטֶלָה	07.07.04
patch [v]	הִטְלִיא	07.07.20
assembler directive	הַנְחָיַת מְאַסֵּף	07.04.67
compiler directive	הַנְחָיַת מְהַדֵּר	07.04.68
interpreter directive	הַנְחָיַת מְפָרֵשׁ	07.04.69
translator directive	הַנְחָיַת מְתַרְגֵּם	07.04.66
address modification	הַסְגָּלַת מַעַן	07.09.41
relocation offset	הֶסֵּט הֲזָחָה	07.05.17
address offset	הֶסֵּט מַעַן	07.05.18
propagate an exception (v)	הֶעֱבִיר חָרִיג	07.06.49
initiate a breakpoint (v)	הִפְסִיק	07.06.22
activation [in computer programming]	הַפְעָלָה	07.10.02
set a breakpoint (v)	הִצִּיב מִפְסָק	07.06.21
segmentation	הַקְטָעָה	07.05.19
starvation	הַרְעָבָה	07.06.38
recover [v]	הִתְאוֹשֵׁשׁ	07.06.33
recovery [in computer programming]	הִתְאוֹשְׁשׁוּת	07.06.34
inline recovery	הִתְאוֹשְׁשׁוּת יְשִׁירָה	07.06.37
backward recovery	הִתְאוֹשְׁשׁוּת לְאָחוֹר	07.06.36
forward recovery	הִתְאוֹשְׁשׁוּת לְפָנִים	07.06.35
hash clash, collision (in hashing)	הִתְנַגְּשׁוּת גִּבּוּב	07.02.21
raise an exception (v)	הִתְרִיעַ עַל חָרִיג	07.06.46
relocatable	זָחִיחַ	07.05.14
assembly time (1)	זְמַן אִסּוּף (1)	07.04.62
execution time (1)	זְמַן בִּצּוּעַ (1)	07.06.02
compilation time (1)	זְמַן הִדּוּר (1)	07.04.61
binding time	זְמַן כְּרִיכָה	07.02.26
processor time	זְמַן מְעַבֵּד	07.06.05
run time (1)	זְמַן רִיצָה (1)	07.06.02
elapsed time	זְמַן שָׁעוֹן	07.06.04
wall-clock time	זְמַן שָׁעוֹן	07.06.04
translation time (1)	זְמַן תִּרְגּוּם (1)	07.04.60
data flow	זְרִימַת נְתוּנִים	07.02.32
bug seeding	זְרִיעַת תַּקָּלוֹת	07.07.32
error seeding	זְרִיעַת תַּקָּלוֹת	07.07.32
fault seeding	זְרִיעַת תַּקָּלוֹת	07.07.32
iteration	חִזְרוּר	07.03.01
fixed-count iteration	חִזְרוּר קָבוּעַ-מִסְפָּר	07.03.10
address part	חֵלֶק מַעַן	07.09.08

English Term	מונח עברי	מספר מונח
operation part, operation field	חֵלֶק פְּעֻלָּה	07.09.06
blocked	חָסוּם	07.10.05
exception	חָרִיג	07.06.45
overflow exception	חֲרִיג גְּלִישָׁה	07.06.54
protection exception	חֲרִיג הֲגָנָה	07.06.53
underflow exception	חֲרִיג חֲמִיקָה	07.06.55
addressing exception	חֲרִיג מְעִינָה	07.06.50
data exception	חֲרִיג נְתוּנִים	07.06.51
operation exception	חֲרִיג פְּעוּלָה	07.06.52
patch	טְלַאי	07.07.19
load [v] [in computer programming]	טָעַן	07.05.05
load-and-go	טְעַן וּבַצֵּעַ	07.05.09
loop assertion	טַעֲנַת לוּלָאָה	07.07.22
assertion	טַעֲנַת תְּנַאי	07.07.21
loop assertion	טַעֲנַת תְּנַאי, קְבִיעָה	07.03.05
handle an exception (v)	טִפֵּל בְּחָרִיג	07.06.48
compilation unit	יְחִידַת הִדּוּר	07.04.22
generic unit	יְחִידַת יְסוֹד, יְחִידָה גֶּנֶרִית	07.04.72
efficiency	יְעִילוּת	07.12.02
exit [v]	יָצָא	07.06.16
software tool	כְּלִי תָּכְנָה (תוכנה)	07.04.47
entrance	כְּנִיסָה	07.06.18
task entry	כְּנִיסַת מְשִׂימָה	07.10.12
binding	כְּרִיכָה	07.02.25
dynamic binding	כְּרִיכָה דִּינָמִית	07.02.28
late binding	כְּרִיכָה מְאֻחֶרֶת	07.02.30
early binding	כְּרִיכָה מֻקְדֶּמֶת	07.02.29
static binding	כְּרִיכָה סְטָטִית	07.02.27
bind [v]	כָּרַךְ	07.02.24
machine-independent	לֹא תְּלוּי-מְכוֹנָה	07.04.36
loop	לוּלָאָה	07.03.03
infinite loop	לוּלָאָה אֵינְסוֹפִית	07.03.04
closed loop	לוּלָאָה סְגוּרָה	07.03.04
cohesion	לְכִידוּת	07.12.05
temporal cohesion	לְכִידוּת בַּזְּמַן	07.12.09
logical cohesion	לְכִידוּת לוֹגִית	07.12.10
informational cohesion	לְכִידוּת מֵידָעִית	07.12.07
coincidental cohesion	לְכִידוּת מִקְרִית	07.12.11
procedural cohesion	לְכִידוּת נְהֵלִית (נוֹהוֹלִית)	07.12.12
sequential cohesion	לְכִידוּת עוֹקֶבֶת	07.12.13
functional cohesion	לְכִידוּת תִּפְקוּדִית	07.12.06
communicational cohesion	לְכִידוּת תִּקְשָׁרְתִּית	07.12.08

English Term	מונח עברי	מספר מונח
	(תקשורתית)	
assembler	מְאַסֵּף	07.04.09
relocating assembler	מְאַסֵּף הֶזָּחָה	07.04.17
absolute assembler	מְאַסֵּף מֻחְלָט	07.04.10
macroassembler	מְאַסֵּף מַקְרוֹ	07.04.77
cross-assembler	מְאַסֵּף צוֹלֵב	07.04.16
microcode assembler	מְאַסֵּף קוֹד-מִיקְרוֹ	07.08.05
continuation test	מִבְחַן הֶמְשֵׁךְ	07.03.12
posttest loop	מִבְחָן לְאַחַר לוּלָאָה	07.03.14
termination test	מִבְחַן סִיּוּם	07.03.11
pretest loop	מִבְחַן-קֶדֶם שֶׁל לוּלָאָה	07.03.13
in-test loop	מִבְחַן-תָּוֶךְ שֶׁל לוּלָאָה	07.03.15
confirmability	מִדַּד אִשּׁוּר	07.12.23
relocation dictionary	מִדְרִיךְ הֶזָּחָה	07.05.16
compiler	מְהַדֵּר	07.04.20
cross-compiler	מְהַדֵּר צוֹלֵב	07.04.25
root compiler	מְהַדֵּר שׁ״רֶשׁ	07.04.41
understandability	מוּבָנוּת	07.12.03
load module	מוֹדוּל טְעִינָה	07.05.08
object module	מוֹדוּל יַעַד	07.04.58
source module	מוֹדוּל מָקוֹר	07.04.39
modularity	מוֹדוּלָרִיּוּת	07.12.04
ready	מוּכָן	07.10.06
execution monitor	מוֹנִיטוֹר בִּצּוּעַ	07.06.15
fork	מַזְלֵג	07.11.04
application generator	מְחוֹלֵל יִשּׁוּמִים	07.04.46
compiler generator, compiler compiler, metacompiler	מְחוֹלֵל מְהַדְּרִים	07.04.24
macrogenerator	מְחוֹלֵל מַקְרוֹ	07.04.73
code generator	מְחוֹלֵל קוֹד	07.04.42
source code generator	מְחוֹלֵל קוֹד-מָקוֹר	07.04.43
program generator	מְחוֹלֵל תָּכְנִיּוֹת (תוכניות)	07.04.78
indicator	מַחֲוָן	07.02.12
host machine (1)	מַחְשֵׁב מְאָרֵחַ (1)	07.04.53
host machine (2)	מַחְשֵׁב מְאָרֵחַ (2)	07.04.54
host machine (3)	מַחְשֵׁב מְאָרֵחַ (3)	07.04.55
microprogrammable computer	מַחְשֵׁב מְבֻקַּר-תָּכְנִית-מִיקְרוֹ	07.08.07
target machine (1)	מַחְשֵׁב מַטָּרָה (1)	07.04.49
target machine (2)	מַחְשֵׁב מַטָּרָה (2)	07.04.50
metalanguage	מֵטָא-שָׂפָה	07.01.02
master task	מַטְלַת אָב	07.10.11
loader	מַטְעָן	07.05.04

English Term	מונח עברי	מספר מונח
bootsrap loader	מַטְעֵן אִתְחוּל	07.06.43
relocating loader	מַטְעֵן זָחִיחַ	07.05.15
absolute loader	מַטְעֵן מֻחְלָט	07.05.06
linking loader	מַטְעֵן מְקַשֵּׁר	07.05.07
exception handler	מְטַפֵּל בַּחֲרִיגִים	07.06.47
operation code trap	מַלְכּוֹד קוֹד פְּעֻלָּה	07.07.17
mutual exclusion	מְנִיעָה הֲדָדִית	07.02.16
fan-out	מְנִיפָה מִתְבַּדֶּרֶת	07.12.21
fan-in	מְנִיפָה מִתְכַּנֶּסֶת	07.12.20
debugger	מְנַפֶּה	07.07.02
modifiability	מֵסְגֻלוּת	07.12.01
macroprocessor	מְעַבֵּד מַקְרוֹ	07.04.74
preprocessor	מְעַבֵּד קֶדֶם	07.04.79
language preprocessor	מְעַבֵּד קֶדֶם שֶׁל שָׂפָה	07.04.81
language processor	מְעַבֵּד שָׂפָה	07.06.01
hashing, hash addressing	מְעִינָה מְגֻבֶּבֶת, גִּבּוּב	07.02.18
implicit addressing, implied addressing	מְעִינָה מִשְׁתַּמַּעַת	07.09.18
repetitive addressing	מְעִינָה נִשְׁנִית	07.09.20
one-ahead addressing	מְעִינָה עוֹקֶבֶת	07.09.19
address	מַעַן	07.09.07
effective address	מַעַן בְּפֹעַל	07.09.40
hash value	מַעַן גִּבּוּב	07.02.20
virtual address	מַעַן וִירְטוּאָלִי	07.09.42
relocatable address	מַעַן זָחִיחַ	07.09.38
relative address	מַעַן יַחֲסִי	07.09.36
self-relative address	מַעַן יַחֲסִי-לְעַצְמוֹ	07.09.46
direct address	מַעַן יָשִׁיר	07.09.33
absolute address	מַעַן מֻחְלָט	07.09.35
generated address	מַעַן מְחֻשָּׁב	07.09.39
indexed address	מַעַן מְמֻפְתָּח	07.09.45
real address	מַעַן מַמָּשִׁי	07.09.43
symbolic address	מַעַן סִמְלִי	07.09.32
multilevel address	מַעַן עָקִיף	07.09.37
indirect address	מַעַן עָקִיף	07.09.37
base address	מַעַן-בָּסִיס	07.09.34
calling sequence	מַעֲקַבֶּת קְרִיאָה	07.09.30
programming system	מַעֲרֶכֶת תִּכְנוּת	07.11.03
breakpoint	מִפְסָק	07.06.20
dynamic breakpoint	מִפְסָק דִּינָמִי	07.06.25
postamble breakpoint	מִפְסַק יְצִיאָה	07.06.29
preamble breakpoint	מִפְסַק כְּנִיסָה	07.06.28
data breakpoint	מִפְסַק נְתוּנִים	07.06.24

English Term	מונח עברי	מספר מונח
static breakpoint	מִפְסָק סְטָטִי	07.06.26
code breakpoint	מִפְסָק קוֹד	07.06.23
control breakpoint	מִפְסָק קוֹד	07.06.23
programmable breakpoint	מִפְסָק תָּכְנִית (תוכנית) נִפּוּי	07.06.27
overlay supervisor	מְפַקֵּחַ רְבָדִים	07.05.23
formal specification	מִפְרָט פּוֹרְמָלִי	07.07.29
parser	מְפָרֵק	07.04.45
interpreter, interpretive program	מְפָרֵשׁ, תָּכְנִית פֵּירוּשׁ	07.04.31
load map	מַפַּת טְעִינָה	07.05.11
task state	מַצַּב מַטָּלָה~מַצַּב מְשִׂימָה	07.10.01
heap	מִצְבּוֹר	07.02.31
segment [in computer programming]	מִקְטָע	07.05.20
overlay segment	מִקְטַע רֹבֶד	07.05.21
address space	מֶרְחַב מְעָנִים, מֶרְחַב מְעִינָה	07.09.31
delayed	מֻשְׁהֶה	07.10.08
master task	מְשִׂימַת אָב	07.10.11
assembly duration' assembly time (2)	מֶשֶׁךְ אִסּוּף	07.04.65
execution time (2)	מֶשֶׁךְ בִּצּוּעַ	07.06.03
run time (2)	מֶשֶׁךְ בִּצּוּעַ	07.06.03
running time	מֶשֶׁךְ בִּצּוּעַ	07.06.03
compilation duration, compilation time (2)	מֶשֶׁךְ הִדּוּר	07.04.64
translation duration, translation time (2)	מֶשֶׁךְ תִּרְגּוּם	07.04.63
loop-control variable	מִשְׁתַּנֶּה בַּקָּרַת לוּלָאָה	07.03.08
shared variable	מִשְׁתַּנֶּה מְשֻׁתָּף	07.02.23
switch	מֶתֶג	07.02.14
programmer	מְתַכְנֵת	07.04.01
translator	מְתַרְגֵּם [שם עצם]	07.04.07
partial correctness	נְכוֹנוּת חֶלְקִית	07.07.30
total correctness	נְכוֹנוּת כּוֹלֶלֶת	07.07.31
indirectly recursive	נָסוֹג בַּעֲקִיפִין	07.03.18
directly recursive	נָסוֹג יְשִׁירוֹת	07.03.17
recursion	נְסִיגָה	07.03.16
simultaneous recursion	נְסִיגָה הֲדָדִית	07.03.19
lockout	נְעִילָה	07.06.40
debug [v]	נִפָּה	07.07.01
checkpoint	נְקֻדַּת בִּקֹּרֶת	07.06.30
reentry point	נְקֻדַּת חֲזָרָה	07.06.19
exit point	נְקֻדַּת יְצִיאָה	07.06.17
entry point	נְקֻדַּת כְּנִיסָה	07.06.18
rescue point	נְקֻדַּת שִׁתְּחוּל	07.06.32
restart point	נְקֻדַּת שִׁתְּחוּל	07.06.32
completed	נִשְׁלַם	07.10.09

English Term	מונח עברי	מספר מונח
immediate data	נָתוּן מִיָּדִי	07.09.23
parser	נַתָּח תַּחְבִּירִי	07.04.45
environment	סְבִיבָה	07.04.02
interactive environment	סְבִיבָה הִדּוּדִית	07.11.10
real-time environment	סְבִיבַת זְמַן-אֱמֶת	07.11.11
batch processing environment	סְבִיבַת עִבּוּד-אֲצָווֹת	07.11.09
programming environment, programming support environment	סְבִיבַת תִּכְנוּת	07.04.03
integrated programming environment, IPE] abb]	סְבִיבַת תִּכְנוּת כְּלוּלִית	07.04.04
operating environment	סְבִיבַת תִּפְעוּל	07.11.08
iteration scheme	סְכֵמַת חִזְרוּר	07.03.09
synchronization	סִנְכְּרוּן	07.02.17
system library	סִפְרִיַּת מַעֲרֶכֶת	07.11.07
macro library	סִפְרִיַּת מַקְרוֹ	07.04.76
software library	סִפְרִיַּת תָּכְנָה	07.11.06
program library	סִפְרִיַּת תָּכְנִיּוֹת	07.11.05
preprocessing	עִבּוּד קֶדֶם	07.04.80
stepwise refinement	עִדּוּן צַעַד-צַעַד	07.02.05
linkage editor, linker	עוֹרֵךְ קִשּׁוּרִים, מְקַשֵּׁר	07.05.03
call tree	עֵץ קְרִיאָה	07.09.48
trace [v]	עָקַב	07.06.08
trace	עִקְבָּה	07.06.07
retrospective trace	עִקְבָּה הִיסְטוֹרִית	07.06.10
execution trace	עִקְבַת בִּצּוּעַ	07.06.09
variable trace	עִקְבַת מִשְׁתַּנִּים	07.06.14
data trace	עִקְבַת נְתוּנִים	07.06.14
data-flow trace	עִקְבַת נְתוּנִים	07.06.14
symbolic trace	עִקְבַת סְמָלִים	07.06.12
subprogram trace	עִקְבַת שְׁגָרוֹת	07.06.11
hash value	עֵרֶךְ גִּבּוּב	07.02.20
scaffolding	פִּגּוּם	07.11.02
hash function (in hashing)	פּוּנְקְצִיַּת גִּבּוּב	07.02.19
pseudocode	פְּסֵידוֹ-קוֹד	07.01.42
microoperation	פְּעֻלַּת מִיקְרוֹ	07.08.06
single-step operation	פְּעֻלַּת צַעַד-צַעַד	07.07.14
step-by-step operation	פְּעֻלַּת צַעַד-צַעַד	07.07.14
instruction	פְּקֻדָּה	07.09.01
direct instruction	פְּקֻדָּה יְשִׁירָה	07.09.21
zero-address instruction	פְּקֻדָּה לְלֹא מַעַן	07.09.12
immediate instruction	פְּקֻדָּה מִיָּדִית	07.09.21
privileged instruction	פְּקֻדָּה מְיֻחֶסֶת	07.09.26

English Term	מונח עברי	מספר מונח
indirect instruction	פְּקֵדָה עֲקִיפָה	07.09.24
n-address instruction	פְּקֵדָה רַבַּת-מְעָנִים	07.09.16
microinstruction	פְּקֵדַת מִיקְרוֹ	07.08.01
machine instruction	פְּקֵדַת מְכוֹנָה	07.09.02
one-plus-one address instruction	פְּקֵדַת מַעַן אֶחָד-וְעוֹד-אֶחָד	07.09.17
one-address instruction	פְּקֵדַת מַעַן יָחִיד	07.09.13
single-address instruction	פְּקֵדַת מַעַן יָחִיד	07.09.13
single-operand instruction	פְּקֵדַת מַעַן יָחִיד	07.09.13
no-op, no-operation instruction	פְּקֵדַת סְרָק	07.09.25
jump instruction	פְּקֵדַת קְפִיצָה	07.09.27
unconditional jump instruction	פְּקֵדַת קְפִיצָה לֹא-מֻתְנֵית	07.09.28
conditional jump instruction	פְּקֵדַת קְפִיצָה מֻתְנֵית	07.09.29
three-address instruction	פְּקֵדַת שְׁלֹשָׁה מְעָנִים	07.09.15
double-operand instruction	פְּקֵדַת שְׁנֵי מְעָנִים	07.09.14
two-address instruction	פְּקֵדַת שְׁנֵי מְעָנִים	07.09.14
execution profile	פְּרוֹפִיל בִּצּוּעַ	07.06.06
loop parameter	פָּרָמֶטֶר לוּלָאָה	07.03.08
parse [v]	פֵּרֵק	07.04.44
interpret [v]	פֵּרֵשׁ	07.04.30
collision resolution (in hashing)	פִּתְרוֹן הִתְנַגְּשׁוּת גִּבּוּב	07.02.22
index [in computer programming]	צִיּוּן	07.09.44
coupling	צִמּוּד	07.12.14
control coupling	צִמּוּד בַּקָּרָה	07.12.16
external coupling	צִמּוּד חִיצוֹנִי	07.12.17
common-environment coupling, common coupling	צִמּוּד מְשֻׁתָּף	07.12.18
data coupling	צִמּוּד נְתוּנִים	07.12.15
content coupling	צִמּוּד תֹּכֶן	07.12.19
iteration step	צַעַד חִזָּרוּר	07.03.02
invariant]adj[	קָבוּעַ [שם תואר], מַשְׁתַּמֵּר	07.07.23
loop invariant	קָבוּעַ לוּלָאָה	07.07.24
assertion	קְבִיעָה	07.07.21
coding [in computer programming]	קִדּוּד	07.04.12
coding [in computer programming]	קוֹד	07.04.11
object code	קוֹד יַעַד	07.04.57
assembly code	קוֹד מְאַסֵּף	07.04.14
compiler code	קוֹד מְהַדֵּר	07.04.23
absolute code	קוֹד מֻחְלָט	07.04.13
microcode	קוֹד מִיקְרוֹ	07.08.04
machine code	קוֹד מְכוֹנָה	07.04.33
source code	קוֹד מָקוֹר	07.04.38
interpretive code	קוֹד פֵּירוּשׁ	07.04.32

English Term	מונח עברי	מספר מונח
operation code	קוֹד פְּעֻלָּה	07.09.11
instruction code	קוֹד פְּקֻדָּה	07.09.10
module strength	קוֹהֶזְיָה	07.12.05
nest [v]	קִנֵּן	07.02.06
deadlock, deadly embrace	קִפָּאוֹן	07.06.39
jump	קְפִיצָה	07.02.10
jump [v]	קָפַץ	07.02.11
inkage, linkl	קִשּׁוּר [שם עצם]	07.05.02
link [v]	קִשֵּׁר	07.05.01
overlay [v]	רִבֵּד	07.05.22
reentrant	רַב-כְּנִיסוֹת	07.03.20
stub	רְכִיב חֲלוּפִי	07.11.01
running	רָץ	07.10.07
recursion	רֵקוּרְסִיָּה	07.03.16
activation record	רְשׁוּמַת הַפְּעֻלָּה	07.05.26
utility routine	שִׁגְרַת שֵׁרוּת	07.11.13
guard	שׁוֹעֵר	07.10.13
closed guard	שׁוֹעֵר סָגוּר	07.10.15
open guard	שׁוֹעֵר פָּתוּחַ	07.10.14
disassemble [v]	שִׁחְזֵר לִשְׂפַת-סַף	07.04.27
decompile [v]	שִׁחְזֵר לִשְׂפַת-עָל	07.04.28
work area	שֶׁטַח עֲבוֹדָה	07.02.15
work space	שֶׁטַח עֲבוֹדָה	07.02.15
working area	שֶׁטַח עֲבוֹדָה	07.02.15
working space	שֶׁטַח עֲבוֹדָה	07.02.15
language [in computer programming]	שָׂפָה	07.01.01
algebraic language	שָׂפָה אַלְגֶּבְּרִית	07.01.17
conversational language	שָׂפָה הִדּוּדִית	07.01.50
interactive language	שָׂפָה הִדּוּדִית	07.01.50
special-purpose language	שָׂפָה יְעוּדָה	07.01.28
problem-oriented language	שָׂפָה כְּוֻנַּת-בְּעָיָה	07.01.18
application-oriented language	שָׂפָה כְּוֻנַּת-יִשּׁוּם	07.01.18
computer-oriented language	שָׂפָה כְּוֻנַּת-מַחְשֵׁב	07.01.05
machine-oriented language	שָׂפָה כְּוֻנַּת-מַחְשֵׁב	07.01.05
object-oriented language	שָׂפָה כְּוֻנַּת-עֶצֶם	07.01.19
general-purpose language	שָׂפָה כְּלָלִית	07.01.27
nonprocedural language	שָׂפָה לֹא נֶהֱלִית	07.01.23
host language	שָׂפָה מְאָרַחַת	07.04.52
extensible language	שָׂפָה מִתְרַחֶבֶת	07.01.16
procedural language	שָׂפָה נֶהֱלִית	07.01.22
procedure-oriented language	שָׂפָה נֶהֱלִית	07.01.22
symbolic language	שָׂפָה סִמְלִית	07.01.10

English Term	מונח עברי	מספר מונח
high-level language	שָׂפָה עִלִּית, שְׂפַת-עָל	07.01.11
high-order language	שָׂפָה עִלִּית, שְׂפַת-עָל	07.01.11
HOL [abb]	שָׂפָה עִלִּית, שְׂפַת-עָל	07.01.11
functional language	שָׂפָה פּוּנְקְצְיוֹנָלִית	07.01.24
elaboration	שִׁפְרוּט	07.10.03
algorithmic language	שְׂפַת אַלְגּוֹרִיתְמִים	07.01.03
test language	שְׂפַת בְּדִיקָה	07.01.44
expression language	שְׂפַת בִּטּוּיִים	07.01.30
intermediate language	שְׂפַת בֵּינַיִם	07.04.40
block-structured language	שְׂפַת גּוּשׁ מֻבְנֶה	07.01.26
1GL [abb]	שְׂפַת דּוֹר רִאשׁוֹן	07.01.09
first-generation language	שְׂפַת דּוֹר רִאשׁוֹן	07.01.09
4GL [abb]	שְׂפַת דּוֹר רְבִיעִי	07.01.14
fourth-generation language	שְׂפַת דּוֹר רְבִיעִי	07.01.14
3GL]abb]	שְׂפַת דּוֹר שְׁלִישִׁי	07.01.13
third-generation language	שְׂפַת דּוֹר שְׁלִישִׁי	07.01.13
2GL [abb]	שְׂפַת דּוֹר שֵׁנִי	07.01.12
second-generation language	שְׂפַת דּוֹר שֵׁנִי	07.01.12
declarative language	שְׂפַת הַכְרָזָה	07.01.20
object language	שְׂפַת יַעַד	07.04.56
authoring language	שְׂפַת לִמְדָּה (לומדה)	07.01.34
target language	שְׂפַת מַטָּרָה	07.04.48
machine language	שְׂפַת מְכוֹנָה	07.01.06
specification language	שְׂפַת מִפְרָט	07.01.37
requirement specification language	שְׂפַת מִפְרַט דְּרִישׁוֹת	07.01.38
compiler specification language	שְׂפַת מִפְרַט מְהַדְּרִים	07.01.43
source language	שְׂפַת מָקוֹר	07.04.34
macrolanguage (1)	שְׂפַת מַקְרוֹ (1)	07.01.35
macrolanguage (2)	שְׂפַת מַקְרוֹ (2)	07.01.36
markup language	שְׂפַת סִימָנֵי עֲרִיכָה	07.01.32
list processing language	שְׂפַת עִבּוּד רְשִׁימוֹת	07.01.29
imperative language	שְׂפַת צִוּוּי	07.01.21
page description language	שְׂפַת תֵּאוּר עַמּוּד	07.01.33
PDL [abb]	שְׂפַת תֵּאוּר עַמּוּד	07.01.33
design language	שְׂפַת תֶּכֶן	07.01.39
hardware design language	שְׂפַת תֶּכֶן חֹמְרָה	07.01.40
HDL [abb]	שְׂפַת תֶּכֶן חֹמְרָה	07.01.40
PDL [abb]	שְׂפַת תֶּכֶן תָּכְנָה	07.01.41
program design language	שְׂפַת תֶּכֶן תָּכְנָה	07.01.41
programming language	שְׂפַת תִּכְנוּת	07.01.04
structured programming language	שְׂפַת תִּכְנוּת מֻבְנֶה	07.01.25
text-formatting language	שְׂפַת תִּסְדוּר, שְׂפַת תִּסְדוּר תַּמְלִיל	07.01.31

English Term	מונח עברי	מספר מונח
assembly language	שְׂפַת-סַף	07.01.08
restart [v]	שִׁתְחֵל	07.06.31
thread	תַּהֲלִיכוֹן	07.10.16
resident]adj[	תּוֹשָׁב [שם תואר]	07.05.24
error prediction	תַּחֲזִית תַּקָּלוֹת	07.07.35
structured design	תָּכְן מֻבְנֶה	07.02.04
logic programming	תִּכְנוּת לוֹגִי	07.02.09
structured programming (1)	תִּכְנוּת מֻבְנֶה (1)	07.02.01
structured programming (2)	תִּכְנוּת מֻבְנֶה (2)	07.02.02
modular programming	תִּכְנוּת מוֹדוּלָרִי	07.02.08
microprogramming	תִּכְנוּת מִיקְרוֹ	07.08.02
macroprogramming	תִּכְנוּת מַקְרוֹ	07.04.75
functional programming	תִּכְנוּת פּוּנְקְצִיוֹנָלִי	07.02.07
diagnostic program	תָּכְנִית (תוכנית) אִבְחוּן	07.07.15
error control software	תָּכְנִית (תוכנית) בַּקָּרַת תַּקָּלוֹת	07.07.34
relocatable program	תָּכְנִית (תוכנית) זְחִיחָה	07.05.13
object program	תָּכְנִית (תוכנית) יַעַד	07.04.59
structured program	תָּכְנִית (תוֹכנית) מֻבְנִית	07.02.03
target program	תָּכְנִית (תוכנית) מַטָּרָה	07.04.51
trace program	תָּכְנִית (תוכנית) מַעֲקָב	07.07.16
source program	תָּכְנִית (תוכנית) מָקוֹר	07.04.37
decompiler	תָּכְנִית (תוֹכנית) שִׁחְזוּר לִשְׂפַת-עָל	07.04.29
utility program	תָּכְנִית (תוֹכנית) שֵׁרוּת	07.11.12
resident program	תָּכְנִית (תוכנית) תּוֹשָׁבֶת	07.05.25
bootstrap	תָּכְנִית אִתְחוּל	07.06.41
initial program load, IPL [abb]	תָּכְנִית אִתְחוּל	07.06.41
IPL [abb], initial program load	תָּכְנִית אִתְחוּל	07.06.41
checking program	תָּכְנִית הַגָּהָה, תָּכְנִית בִּדּוּק	07.07.18
microprogram	תָּכְנִית מִיקְרוֹ	07.08.03
debugger	תָּכְנִית נִפּוּי	07.07.02
translation program	תָּכְנִית תִּרְגּוּם	07.04.07
programmer	תָּכְנִיתָן	07.04.01
programmer	תַּכְנָת	07.04.01
machine-dependent	תְּלוּי-מְכוֹנָה	07.04.35
postcondition	תְּנַאי-אַחַר	07.07.26
precondition	תְּנַאי-קֶדֶם	07.07.25
address format	תַּסְדִּיר מַעַן	07.09.09
instruction format	תַּסְדִּיר פְּקֻדָּה	07.09.03
indigenous error, indigenous fault	תַּקָּלָה יְלִידָה	07.07.33
unrecoverable error	תַּקָּלָה לְלֹא תַּקָּנָה	07.07.36
translation	תִּרְגּוּם	07.04.06
translate [v]	תִּרְגֵּם	07.04.05

English Term	מונח עברי	מספר מונח
bubble chart	תַּרְשִׁים בּוּעוֹת	07.09.52
state transition diagram, state diagram	תַּרְשִׁים הֶחְלֵף מַצָּבִים, תַּרְשִׁים מַצָּבִים	07.09.54
control flow diagram, control flow graph	תַּרְשִׁים זְרִימַת בַּקָּרָה	07.09.49
data flow diagram, data flowchart, data flow graph	תַּרְשִׁים זְרִימַת נְתוּנִים	07.09.51
structure chart	תַּרְשִׁים מִבְנֶה	07.09.47
hierarchy chart	תַּרְשִׁים מִדְרָג	07.09.47
IPO chart	תַּרְשִׁים קֶלֶט-עִבּוּד-פֶּלֶט	07.09.53
input-process-output chart	תַּרְשִׁים קֶלֶט-עִבּוּד-פֶּלֶט	07.09.53
call graph	תַּרְשִׁים קְרִיאָה	07.09.48
box diagram	תַּרְשִׁים תֵּבוֹת	07.09.50
Chapin chart	תַּרְשִׁים תֵּבוֹת	07.09.50
Nassi-Shneiderman chart	תַּרְשִׁים תֵּבוֹת	07.09.50

מפתח מונחים A-B

English Term	מונח עברי	מספר מונח
1GL [abb]	שְׂפַת דּוֹר רִאשׁוֹן	07.01.09
2GL [abb]	שְׂפַת דּוֹר שֵׁנִי	07.01.12
3GL]abb]	שְׂפַת דּוֹר שְׁלִישִׁי	07.01.13
4GL [abb]	שְׂפַת דּוֹר רְבִיעִי	07.01.14
absolute address	מַעַן מֻחְלָט	07.09.35
absolute assembler	מְאַסֵּף מֻחְלָט	07.04.10
absolute code	קוֹד מֻחְלָט	07.04.13
absolute loader	מַטְעַן מֻחְלָט	07.05.06
activation [in computer programming]	הַפְעָלָה	07.10.02
activation record	רְשׁוּמַת הַפְעָלָה	07.05.26
address	מַעַן	07.09.07
address format	תַּסְדִּיר מַעַן	07.09.09
address modification	הַסְגָּלַת מַעַן	07.09.41
address offset	הֶסֵּט מַעַן	07.05.18
address part	חֵלֶק מַעַן	07.09.08
address space	מֶרְחַב מְעָנִים, מֶרְחַב מְעִינָה	07.09.31
addressing exception	חֲרִיג מְעִינָה	07.06.50
algebraic language	שָׂפָה אַלְגֶּבְּרִית	07.01.17
algorithmic language	שְׂפַת אַלְגּוֹרִיתְמִים	07.01.03
application generator	מְחוֹלֵל יִשּׁוּמִים	07.04.46
application-oriented language	שָׂפָה כְּוֻנַּת-יִשּׁוּם	07.01.18
assemble [v]	אִסֵּף	07 04.08
assemble-and-go	אַסֵּף וּבַצַּע	07.04.18
assembled origin	בְּסִיס-סַף	07.04.15
assembler	מְאַסֵּף	07.04.09
assembler directive	הַנְחָיַת מְאַסֵּף	07.04.67
assembly code	קוֹד מְאַסֵּף	07.04.14
assembly duration' assembly time (2)	מֶשֶׁךְ אִסּוּף	07.04.65
assembly language	שְׂפַת-סַף	07.01.08
assembly time (1)	זְמַן אִסּוּף (1)	07.04.62
assertion	טַעֲנַת תְּנַאי	07.07.21
assertion	קְבִיעָה	07.07.21
authoring language	שְׂפַת לִמְדָּה (לומדה)	07.01.34
backward recovery	הִתְאוֹשְׁשׁוּת לְאָחוֹר	07.06.36
base address	מַעַן-בָּסִיס	07.09.34
batch processing environment	סְבִיבַת עִבּוּד-אֲצָוֹות	07.11.09
bind [v]	כָּרַךְ	07.02.24
binding	כְּרִיכָה	07.02.25

English Term	מונח עברי	מספר מונח
binding time	זְמַן כְּרִיכָה	07.02.26
blocked	חָסוּם	07.10.05
block-structured language	שְׂפַת גּוּשׁ מֻבְנֶה	07.01.26
boot [v]	אִתְחֵל (2)	07.06.44
bootsrap loader	מַטְעֵן אִתְחוּל	07.06.43
bootstrap	תָּכְנִית אִתְחוּל	07.06.41
bootstrap [v]	אִתְחֵל (1)	07.06.42
box diagram	תַּרְשִׁים תֵּבוֹת	07.09.50
breakpoint	מִפְסָק	07.06.20
bubble chart	תַּרְשִׁים בּוּעוֹת	07.09.52
bug seeding	זְרִיעַת תַּקָּלוֹת	07.07.32
call graph	תַּרְשִׁים קְרִיאָה	07.09.48
call tree	עֵץ קְרִיאָה	07.09.48
calling sequence	מַעֲקַ֫בֶת קְרִיאָה	07.09.30
change dump	הֶטֵל שִׁנּוּיִים	07.07.07
Chapin chart	תַּרְשִׁים תֵּבוֹת	07.09.50
checking program	תָּכְנִית הַגָּהָה, תָּכְנִית בְּדוּק	07.07.18
checkpoint	נְקֻדַּת בִּקֹּ֫רֶת	07.06.30
closed guard	שׁוֹעֵר סָגוּר	07.10.15
closed loop	לוּלָאָה סְגוּרָה	07.03.04
code breakpoint	מִפְסַק קוֹד	07.06.23
code generator	מְחוֹלֵל קוֹד	07.04.42
coding [in computer programming]	קִדּוּד	07.04.12
coding [in computer programming]	קוֹד	07.04.11
cohesion	לְכִידוּת	07.12.05
coincidental cohesion	לְכִידוּת מִקְרִית	07.12.11
collision resolution (in hashing)	פִּתְרוֹן הִתְנַגְּשׁוּת גִּבּוּב	07.02.22
common-environment coupling, common coupling	צִמּוּד מְשֻׁתָּף	07.12.18
communicational cohesion	לְכִידוּת תִּקְשָׁרְתִּית (תקשורתית)	07.12.08
compilation	הִדּוּר	07.04.21
compilation duration, compilation time (2)	מֶשֶׁךְ הִדּוּר	07.04.64
compilation time (1)	זְמַן הִדּוּר (1)	07.04.61
compilation unit	יְחִידַת הִדּוּר	07.04.22
compile [v]	הִדֵּר	07.04.19
compile-and-go	הַדֵּר וּבַצַּע	07.04.26
compiler	מְהַדֵּר	07.04.20
compiler code	קוֹד מְהַדֵּר	07.04.23
compiler directive	הַנְחָיַת מְהַדֵּר	07.04.68
compiler generator, compiler compiler, metacompiler	מְחוֹלֵל מְהַדְּרִים	07.04.24

English Term	מונח עברי	מספר מונח
compiler specification language	שְׂפַת מִפְרַט מְהַדְּרִים	07.01.43
completed	נִשְׁלַם	07.10.09
computer-oriented language	שָׂפָה כְּוֻנַּת-מַחְשֵׁב	07.01.05
conditional jump instruction	פְּקֻדַּת קְפִיצָה מֻתְנֵית	07.09.29
confirmability	מִדַּד אִשּׁוּר	07.12.23
content coupling	צִמּוּד תֹּכֶן	07.12.19
continuation test	מִבְחַן הֶמְשֵׁךְ	07.03.12
control breakpoint	מִפְסַק קוֹד	07.06.23
control coupling	צִמּוּד בַּקָּרָה	07.12.16
control flow diagram, control flow graph	תַּרְשִׁים זְרִימַת בַּקָּרָה	07.09.49
conversational language	שָׂפָה הִדּוּדִית	07.01.50
correctness proving	הוֹכָחַת נְכוֹנוּת (1) [הפעולה]	07.07.27
coupling	צִמּוּד	07.12.14
cross-assembler	מְאַסֵּף צוֹלֵב	07.04.16
cross-compiler	מְהַדֵּר צוֹלֵב	07.04.25
data breakpoint	מִפְסַק נְתוּנִים	07.06.24
data coupling	צִמּוּד נְתוּנִים	07.12.15
data exception	חֲרִיג נְתוּנִים	07.06.51
data flow	זְרִימַת נְתוּנִים	07.02.32
data flow diagram, data flowchart, data flow graph	תַּרְשִׁים זְרִימַת נְתוּנִים	07.09.51
data trace	עִקְבַת נְתוּנִים	07.06.14
data-flow trace	עִקְבַת נְתוּנִים	07.06.14
deadlock, deadly embrace	קִפָּאוֹן	07.06.39
debug [v]	נִפָּה	07.07.01
debugger	מְנַפֶּה	07.07.02
debugger	תָּכְנִית נִפּוּי	07.07.02
declarative language	שְׂפַת הַכְרָזָה	07.01.20
decompile [v]	שִׁחְזֵר לִשְׂפַת-עָל	07.04.28
decompiler	תָּכְנִית (תוֹכנית) שִׁחְזוּר לִשְׂפַת-עָל	07.04.29
delayed	מֻשְׁהֶה	07.10.08
dependent compilation	הִדּוּר תָּלוּי	07.04.70
design language	שְׂפַת תֶּכֶן	07.01.39
desk checking	בְּדוּק יְדָנִי	07.07.11
diagnostic program	תָּכְנִית (תוכנית) אִבְחוּן	07.07.15
direct address	מַעַן יָשִׁיר	07.09.33
direct instruction	פְּקֻדָּה יְשִׁירָה	07.09.21
directly recursive	נָסוֹג יְשִׁירוֹת	07.03.17
disassemble [v]	שִׁחְזֵר לִשְׂפַת-סַף	07.04.27
double-operand instruction	פְּקֻדַּת שְׁנֵי מְעָנִים	07.09.14
dump (1)	הֶטְלָה	07.07.04
dump (2), data dump	הֶטֵל, הֶטֵל נְתוּנִים	07.07.05

English Term	מונח עברי	מספר מונח
dump [v]	הַטִיל	07.07.03
dynamic binding	כְּרִיכָה דִינָמִית	07.02.28
dynamic breakpoint	מִפְסָק דִינָמִי	07.06.25
early binding	כְּרִיכָה מֻקְדֶּמֶת	07.02.29
effective address	מַעַן בְּפֹעַל	07.09.40
efficiency	יְעִילוּת	07.12.02
elaboration	שִׁפְרוּט	07.10.03
elapsed time	זְמַן שָׁעוֹן	07.06.04
entrance	כְּנִיסָה	07.06.18
entry point	נְקֻדַּת כְּנִיסָה	07.06.18
environment	סְבִיבָה	07.04.02
error control software	תָּכְנִית (תוכנית) בַּקָּרַת תַּקָּלוֹת	07.07.34
error prediction	תַּחֲזִית תַּקָּלוֹת	07.07.35
error seeding	זְרִיעַת תַּקָּלוֹת	07.07.32
exception	חָרִיג	07.06.45
exception handler	מְטַפֵּל בַּחֲרִיגִים	07.06.47
executable	בָּצִיעַ	07.10.04
execution monitor	מוֹנִיטוֹר בִּצּוּעַ	07.06.15
execution profile	פְּרוֹפִיל בִּצּוּעַ	07.06.06
execution time (1)	זְמַן בִּצּוּעַ (1)	07.06.02
execution time (2)	מֶשֶׁךְ בִּצּוּעַ	07.06.03
execution trace	עִקְבַת בִּצּוּעַ	07.06.09
exit [v]	יָצָא	07.06.16
exit point	נְקֻדַּת יְצִיאָה	07.06.17
expression language	שְׂפַת בִּטּוּיִים	07.01.30
extensible language	שָׂפָה מִתְרַחֶבֶת	07.01.16
external coupling	צִמּוּד חִיצוֹנִי	07.12.17
fan-in	מְנִיפָה מִתְכַּנֶּסֶת	07.12.20
fan-out	מְנִיפָה מִתְבַּדֶּרֶת	07.12.21
fault seeding	זְרִיעַת תַּקָּלוֹת	07.07.32
first-generation language	שְׂפַת דּוֹר רִאשׁוֹן	07.01.09
fixed-count iteration	חִזְרוּר קָבוּעַ-מִסְפָּר	07.03.10
flag	דֶּגֶל	07.02.13
fork	מַזְלֵג	07.11.04
formal specification	מִפְרָט פּוֹרְמָלִי	07.07.29
forward recovery	הִתְאוֹשְׁשׁוּת לְפָנִים	07.06.35
fourth-generation language	שְׂפַת דּוֹר רְבִיעִי	07.01.14
functional cohesion	לְכִידוּת תִּפְקוּדִית	07.12.06
functional language	שָׂפָה פּוּנְקְצְיוֹנָלִית	07.01.24
functional programming	תִּכְנוּת פּוּנְקְצְיוֹנָלִי	07.02.07
general-purpose language	שָׂפָה כְּלָלִית	07.01.27
generated address	מַעַן מְחֻשָּׁב	07.09.39

English Term	מונח עברי	מספר מונח
generic unit	יְחִידַת יְסוֹד, יְחִידָה גֶּנֶרִית	07.04.72
guard	שׁוֹעֵר	07.10.13
handle an exception (v)	טִפֵּל בְּחָרִיג	07.06.48
hardware design language	שְׂפַת תֶּכֶן חֲמָרָה	07.01.40
hash clash, collision (in hashing)	הִתְנַגְּשׁוּת גִּבּוּב	07.02.21
hash function (in hashing)	פּוּנְקְצִיַּת גִּבּוּב	07.02.19
hash value	מַעַן גִּבּוּב	07.02.20
hash value	עֵרֶךְ גִּבּוּב	07.02.20
hashing, hash addressing	גִּבּוּב, מְעִינָה מְגֻבֶּבֶת	07.02.18
hashing, hash addressing	מְעִינָה מְגֻבֶּבֶת, גִּבּוּב	07.02.18
HDL [abb]	שְׂפַת תֶּכֶן חֲמָרָה	07.01.40
heap	מִצְבּוֹר	07.02.31
hierarchy chart	תַּרְשִׁים מִדְרָג	07.09.47
high-level language	שָׂפָה עִלִּית, שְׂפַת-עָל	07.01.11
high-order language	שָׂפָה עִלִּית, שְׂפַת-עָל	07.01.11
HOL [abb]	שָׂפָה עִלִּית, שְׂפַת-עָל	07.01.11
host language	שָׂפָה מְאָרַחַת	07.04.52
host machine (1)	מַחְשֵׁב מְאָרֵחַ (1)	07.04.53
host machine (2)	מַחְשֵׁב מְאָרֵחַ (2)	07.04.54
host machine (3)	מַחְשֵׁב מְאָרֵחַ (3)	07.04.55
immediate data	נָתוּן מִיָּדִי	07.09.23
immediate instruction	פְּקֻדָּה מִיָּדִית	07.09.21
immediate operand	אוֹפֶרַנְד מִיָּדִי	07.09.22
imperative language	שְׂפַת צִוּוּי	07.01.21
implicit addressing, implied addressing	מְעִינָה מִשְׁתַּמַּעַת	07.09.18
independent compilation	הִדּוּר לֹא-תָּלוּי	07.04.71
index [in computer programming]	אִינְדֶּקְס	07.09.44
index [in computer programming]	צִיּוּן	07.09.44
indexed address	מַעַן מְמֻפְתָּח	07.09.45
indicator	מַחֲוָן	07.02.12
indigenous error, indigenous fault	תַּקָּלָה יְלִידָה	07.07.33
indirect address	מַעַן עָקִיף	07.09.37
indirect instruction	פְּקֻדָּה עֲקִיפָה	07.09.24
indirectly recursive	נָסוֹג בַּעֲקִיפִין	07.03.18
infinite loop	לוּלָאָה אֵינְסוֹפִית	07.03.04
informational cohesion	לְכִידוּת מֵידָעִית	07.12.07
initial program load, IPL [abb]	תָּכְנִית אִתְחוּל	07.06.41
initiate a breakpoint (v)	הִפְסִיק	07.06.22
inkage, linkl	קִשּׁוּר [שם עצם]	07.05.02
inline recovery	הִתְאוֹשְׁשׁוּת יְשִׁירָה	07.06.37
input-process-output chart	תַּרְשִׁים קֶלֶט-עִבּוּד-פֶּלֶט	07.09.53
instruction	הוֹרָאָה	07.09.01

English Term	מונח עברי	מספר מונח
instruction	פְּקֻדָּה	07.09.01
instruction code	קוֹד פְּקֻדָּה	07.09.10
instruction format	תַּסְדִּיר פְּקֻדָּה	07.09.03
instruction lentgh	אֹרֶךְ פְּקֻדָּה	07.09.05
instruction set, instruction repertoire	אוֹצַר פְּקֻדּוֹת	07.09.04
integrated programming environment, IPE] abb]	סְבִיבַת תִּכְנוּת כְּלוּלִית	07.04.04
interactive environment	סְבִיבָה הִדּוּדִית	07.11.10
interactive language	שָׂפָה הִדּוּדִית	07.01.50
intermediate language	שְׂפַת בֵּינַיִם	07.04.40
interpret [v]	פֵּרֵשׁ	07.04.30
interpreter directive	הַנְחָיַת מְפָרֵשׁ	07.04.69
interpreter, interpretive program	מְפָרֵשׁ, תָּכְנִית פֵּירוּשׁ	07.04.31
interpretive code	קוֹד פֵּירוּשׁ	07.04.32
in-test loop	מִבְחַן-תָּוֶךְ שֶׁל לוּלָאָה	07.03.15
invariant]adj[	קָבוּעַ [שם תואר], מִשְׁתַּמֵּר	07.07.23
IPL [abb], initial program load	תָּכְנִית אִתְחוּל	07.06.41
IPO chart	תַּרְשִׁים קֶלֶט-עִבּוּד-פֶּלֶט	07.09.53
iteration	אִיטֵרַצְיָה	07.03.01
iteration	חִזָּרוּר	07.03.01
iteration scheme	סְכֵמַת חִזָּרוּר	07.03.09
iteration step	צַעַד חִזָּרוּר	07.03.02
jump	קְפִיצָה	07.02.10
jump [v]	קָפַץ	07.02.11
jump instruction	פְּקֻדַּת קְפִיצָה	07.09.27
language [in computer programming]	שָׂפָה	07.01.01
language preprocessor	מְעַבֵּד קֶדֶם שֶׁל שָׂפָה	07.04.81
language processor	מְעַבֵּד שָׂפָה	07.06.01
late binding	כְּרִיכָה מְאֻחֶרֶת	07.02.30
link [v]	קִשֵּׁר	07.05.01
linkage editor, linker	עוֹרֵךְ קִשּׁוּרִים, מְקַשֵּׁר	07.05.03
linking loader	מַטְעָן מְקַשֵּׁר	07.05.07
list processing language	שְׂפַת עִבּוּד רְשִׁימוֹת	07.01.29
load [v] [in computer programming]	טָעַן	07.05.05
load map	מַפַּת טְעִינָה	07.05.11
load module	מוֹדוּל טְעִינָה	07.05.08
load-and-go	טְעַן וּבַצַּע	07.05.09
loaded origin	בְּסִיס טְעִינָה	07.05.10
loader	מַטְעָן	07.05.04
localization	אִתּוּר	07.12.22
lockout	נְעִילָה	07.06.40
logic programming	תִּכְנוּת לוֹגִי	07.02.09

English Term	מונח עברי	מספר מונח
logical cohesion	לְכִידוּת לוֹגִית	07.12.10
loop	לוּלָאָה	07.03.03
loop assertion	טַעֲנַת לוּלָאָה	07.07.22
loop assertion	טַעֲנַת תְּנַאי, קְבִיעָה	07.03.05
loop body	גּוּף הַלּוּלָאָה	07.03.06
loop control	בַּקָּרַת לוּלָאָה	07.03.07
loop invariant	קָבוּעַ לוּלָאָה	07.07.24
loop parameter	פָּרָמֶטֶר לוּלָאָה	07.03.08
loop-control variable	מִשְׁתַּנֶּה בַּקָּרַת לוּלָאָה	07.03.08
machine code	קוֹד מְכוֹנָה	07.04.33
machine instruction	פְּקֻדַּת מְכוֹנָה	07.09.02
machine language	שְׂפַת מְכוֹנָה	07.01.06
machine-dependent	תְּלוּי-מְכוֹנָה	07.04.35
machine-independent	לֹא תְּלוּי-מְכוֹנָה	07.04.36
machine-oriented language	שָׂפָה כְּוֻנַּת-מַחְשֵׁב	07.01.05
macro library	סִפְרִיַּת מַקְרוֹ	07.04.76
macroassembler	מְאַסֵּף מַקְרוֹ	07.04.77
macrogenerator	מְחוֹלֵל מַקְרוֹ	07.04.73
macrolanguage (1)	שְׂפַת מַקְרוֹ (1)	07.01.35
macrolanguage (2)	שְׂפַת מַקְרוֹ (2)	07.01.36
macroprocessor	מְעַבֵּד מַקְרוֹ	07.04.74
macroprogramming	תִּכְנוּת מַקְרוֹ	07.04.75
markup language	שְׂפַת סִימְנֵי עֲרִיכָה	07.01.32
master task	מַטְלַת אָב	07.10.11
master task	מְשִׂימַת אָב	07.10.11
memory dump	הֶטֵל זִכָּרוֹן	07.07.10
metalanguage	מֵטָא-שָׂפָה	07.01.02
microcode	קוֹד מִיקְרוֹ	07.08.04
microcode assembler	מְאַסֵּף קוֹד-מִיקְרוֹ	07.08.05
microinstruction	פְּקֻדַּת מִיקְרוֹ	07.08.01
microoperation	פְּעֻלַּת מִיקְרוֹ	07.08.06
microprogram	תָּכְנִית מִיקְרוֹ	07.08.03
microprogrammable computer	מַחְשֵׁב מְבֻקַּר-תָּכְנִית-מִיקְרוֹ	07.08.07
microprogramming	תִּכְנוּת מִיקְרוֹ	07.08.02
modifiability	מֻסְגָּלוּת	07.12.01
modular programming	תִּכְנוּת מוֹדוּלָרִי	07.02.08
modularity	מוֹדוּלָרִיּוּת	07.12.04
module strength	קוֹהֶזְיָה	07.12.05
multilevel address	מַעַן עָקִיף	07.09.37
mutual exclusion	מְנִיעָה הֲדָדִית	07.02.16
n-address instruction	פְּקֻדָּה רַבַּת-מְעָנִים	07.09.16
Nassi-Shneiderman chart	תַּרְשִׁים תֵּבוֹת	07.09.50

English Term	מונח עברי	מספר מונח
nest [v]	קִנֵּן	07.02.06
nonprocedural language	שָׂפָה לֹא נֶהֱלִית	07.01.23
no-op, no-operation instruction	פְּקֻדַּת סְרָק	07.09.25
object code	קוֹד יַעַד	07.04.57
object language	שְׂפַת יַעַד	07.04.56
object module	מוֹדוּל יַעַד	07.04.58
object program	תָּכְנִית (תוכנית) יַעַד	07.04.59
object-oriented language	שָׂפָה כְּוֻנַּת-עֶצֶם	07.01.19
one-address instruction	פְּקֻדַּת מַעַן יָחִיד	07.09.13
one-ahead addressing	מְעִינָה עוֹקֶבֶת	07.09.19
one-plus-one address instruction	פְּקֻדַּת מַעַן אֶחָד-וְעוֹד-אֶחָד	07.09.17
open guard	שׁוֹעֵר פָּתוּחַ	07.10.14
operating environment	סְבִיבַת תִּפְעוּל	07.11.08
operation code	קוֹד פְּעֻלָּה	07.09.11
operation code trap	מַלְכּוֹד קוֹד פְּעֻלָּה	07.07.17
operation exception	חֲרִיג פְּעוּלָּה	07.06.52
operation part, operation field	חֵלֶק פְּעֻלָּה	07.09.06
overflow exception	חֲרִיג גְּלִישָׁה	07.06.54
overlay [v]	רִבֵּד	07.05.22
overlay segment	מִקְטַע רִבֻּד	07.05.21
overlay supervisor	מְפַקֵּחַ רִבָּדִים	07.05.23
page description language	שְׂפַת תֵּאוּר עַמּוּד	07.01.33
parse [v]	פִּרֵק	07.04.44
parser	מְפָרֵק	07.04.45
parser	נַתָּח תַּחְבִּירִי	07.04.45
partial correctness	נְכוֹנוּת חֶלְקִית	07.07.30
patch	טְלָאי	07.07.19
patch [v]	הִטְלִיא	07.07.20
PDL [abb]	שְׂפַת תֵּאוּר עַמּוּד	07.01.33
PDL [abb]	שְׂפַת תֶּכֶן תְּכְנָה	07.01.41
playback	בִּצּוּעַ מֻקְלָט	07.07.12
postamble breakpoint	מִפְסַק יְצִיאָה	07.06.29
postcondition	תְּנַאי-אַחַר	07.07.26
postmortem dump	הֶטֵל סִיּוּם, הֶטֵל סִיּוּם לֹא-תַּקִּין	07.07.08
posttest loop	מִבְחָן לְאַחַר לוּלָאָה	07.03.14
preamble breakpoint	מִפְסַק כְּנִיסָה	07.06.28
precondition	תְּנַאי-קֶדֶם	07.07.25
preprocessing	עִבּוּד קֶדֶם	07.04.80
preprocessor	מְעַבֵּד קֶדֶם	07.04.79
pretest loop	מִבְחַן-קֶדֶם שֶׁל לוּלָאָה	07.03.13
privileged instruction	פְּקֻדָּה מְיֻחֶסֶת	07.09.26
problem-oriented language	שָׂפָה כְּוֻנַּת-בְּעָיָה	07.01.18

English Term	מונח עברי	מספר מונח
procedural cohesion	לְכִידוּת נָהֳלִית (נוֹהוֹלית)	07.12.12
procedural language	שָׂפָה נָהֳלִית	07.01.22
procedure-oriented language	שָׂפָה נָהֳלִית	07.01.22
processor time	זְמַן מְעַבֵּד	07.06.05
program design language	שְׂפַת תֶּכֶן תְּכָנָה	07.01.41
program generator	מְחוֹלֵל תְּכָנִיּוֹת (תוכניות)	07.04.78
program library	סִפְרִיַּת תְּכָנִיּוֹת	07.11.05
programmable breakpoint	מִפְסַק תְּכָנִית (תוכנית) נִפּוּי	07.06.27
programmer	מְתַכְנֵת	07.04.01
programmer	תְּכָנִיתָן	07.04.01
programmer	תַּכְנָת	07.04.01
programming environment, programming support environment	סְבִיבַת תִּכְנוּת	07.04.03
programming language	שְׂפַת תִּכְנוּת	07.01.04
programming system	מַעֲרֶכֶת תִּכְנוּת	07.11.03
proof of correctness	הוֹכָחַת נְכוֹנוּת (2) [התוצאה]	07.07.28
propagate an exception (v)	הֶעֱבִיר חָרִיג	07.06.49
protection exception	חָרִיג הֶגָּנָה	07.06.53
pseudocode	פְּסֵידוֹ-קוֹד	07.01.42
raise an exception (v)	הִתְרִיעַ עַל חָרִיג	07.06.46
ready	מוּכָן	07.10.06
real address	מַעַן מַמָּשִׁי	07.09.43
real-time environment	סְבִיבַת זְמַן-אֱמֶת	07.11.11
recover [v]	הִתְאוֹשֵׁשׁ	07.06.33
recovery [in computer programming]	הִתְאוֹשְׁשׁוּת	07.06.34
recursion	נְסִיגָה	07.03.16
recursion	רֵקוּרְסִיָּה	07.03.16
reentrant	רַב-כְּנִיסוֹת	07.03.20
reentry point	נְקֵדַת חֲזָרָה	07.06.19
rehashing	גִּבּוּב חוֹזֵר	07.02.22
relative address	מַעַן יַחֲסִי	07.09.36
relocatable	זָחִיחַ	07.05.14
relocatable address	מַעַן זָחִיחַ	07.09.38
relocatable program	תְּכָנִית (תוכנית) זְחִיחָה	07.05.13
relocate [v]	הִזִּיחַ	07.05.12
relocating assembler	מְאַסֵּף הֲזָחָה	07.04.17
relocating loader	מַטְעַן זָחִיחַ	07.05.15
relocation dictionary	מִדְרִיךְ הֲזָחָה	07.05.16
relocation offset	הֶסֵּט הֲזָחָה	07.05.17
repetitive addressing	מְעִינָה נִשְׁנִית	07.09.20
replay	בִּצּוּעַ חוֹזֵר	07.07.13
requirement specification language	שְׂפַת מִפְרַט דְּרִישׁוֹת	07.01.38

English Term	מונח עברי	מספר מונח
rescue point	נְקֻדַּת שִׁתְּחוּל	07.06.32
resident]adj[	תּוֹשָׁב [שם תואר]	07.05.24
resident program	תָּכְנִית (תוכנית) תּוֹשָׁבֶת	07.05.25
restart [v]	שִׁתְּחֵל	07.06.31
restart point	נְקֻדַּת שִׁתְּחוּל	07.06.32
retrospective trace	עִקְבָה הִיסְטוֹרִית	07.06.10
root compiler	מְהַדֵּר שׁוֹרֶשׁ	07.04.41
run time (1)	זְמַן רִיצָה (1)	07.06.02
run time (2)	מֶשֶׁךְ בִּצּוּעַ	07.06.03
running	רָץ	07.10.07
running time	מֶשֶׁךְ בִּצּוּעַ	07.06.03
scaffolding	פִּגּוּם	07.11.02
second-generation language	שְׂפַת דּוֹר שֵׁנִי	07.01.12
segment [in computer programming]	מִקְטָע	07.05.20
segmentation	הַקְטָעָה	07.05.19
selective dump	הֶטֵל בּוֹרֵר	07.07.06
self-relative address	מַעַן יַחֲסִי-לְעַצְמוֹ	07.09.46
separate compilation	הִדּוּר תְּלוּי	07.04.70
sequential cohesion	לְכִידוּת עוֹקֶבֶת	07.12.13
set a breakpoint (v)	הִצִּיב מִפְסָק	07.06.21
shared variable	מִשְׁתַּנֶּה מְשֻׁתָּף	07.02.23
simultaneous recursion	נְסִיגָה הֲדָדִית	07.03.19
single-address instruction	פְּקֻדַּת מַעַן יָחִיד	07.09.13
single-operand instruction	פְּקֻדַּת מַעַן יָחִיד	07.09.13
single-step execution	בִּצּוּעַ צַעַד-צַעַד	07.07.14
single-step operation	פְּעֻלַּת צַעַד-צַעַד	07.07.14
snapshot dump	הֶטֵל בָּזָק	07.07.09
software library	סִפְרִיַּת תָּכְנָה	07.11.06
software tool	כְּלִי תָּכְנָה (תוכנה)	07.04.47
source code	קוֹד מָקוֹר	07.04.38
source code generator	מְחוֹלֵל קוֹד-מָקוֹר	07.04.43
source language	שְׂפַת מָקוֹר	07.04.34
source module	מוֹדוּל מָקוֹר	07.04.39
source program	תָּכְנִית (תוכנית) מָקוֹר	07.04.37
special-purpose language	שָׂפָה יְעוּדָה	07.01.28
specification language	שְׂפַת מִפְרָט	07.01.37
starvation	הַרְעָבָה	07.06.38
state transition diagram, state diagram	תַּרְשִׁים הֶחְלֵף מַצָּבִים, תַּרְשִׁים מַצָּבִים	07.09.54
static binding	כְּרִיכָה סְטָטִית	07.02.27
static breakpoint	מִפְסָק סְטָטִי	07.06.26
step-by-step operation	פְּעֻלַּת צַעַד-צַעַד	07.07.14
stepwise refinement	עִדּוּן צַעַד-צַעַד	07.02.05

English Term	מונח עברי	מספר מונח
structure chart	תַּרְשִׁים מִבְנֶה	07.09.47
structured design	תָּכָן מֻבְנֶה	07.02.04
structured program	תָּכְנִית (תוֹכנית) מֻבְנֵית	07.02.03
structured programming (1)	תִּכְנוּת מֻבְנֶה (1)	07.02.01
structured programming (2)	תִּכְנוּת מֻבְנֶה (2)	07.02.02
structured programming language	שְׂפַת תִּכְנוּת מֻבְנֶה	07.01.25
stub	רְכִיב חֲלוּפִי	07.11.01
subprogram trace	עִקְבַת שְׁגָרוֹת	07.06.11
switch	מֶתֶג	07.02.14
symbolic address	מַעַן סִמְלִי	07.09.32
symbolic execution	בִּצּוּעַ סִמְלִי	07.06.13
symbolic language	שָׂפָה סִמְלִית	07.01.10
symbolic trace	עִקְבַת סְמָלִים	07.06.12
synchronization	סִנְכְּרוּן	07.02.17
system library	סִפְרִיַּת מַעֲרֶכֶת	07.11.07
target language	שְׂפַת מַטָּרָה	07.04.48
target machine (1)	מַחְשֵׁב מַטָּרָה (1)	07.04.49
target machine (2)	מַחְשֵׁב מַטָּרָה (2)	07.04.50
target program	תָּכְנִית (תוכנית) מַטָּרָה	07.04.51
task entry	כְּנִיסַת מְשִׂימָה	07.10.12
task state	מַצַּב מַטָּלָה~מַצַּב מְשִׂימָה	07.10.01
temporal cohesion	לְכִידוּת בַּזְּמַן	07.12.09
terminated	גָּמוּר	07.10.10
termination test	מִבְחַן סִיּוּם	07.03.11
test language	שְׂפַת בְּדִיקָה	07.01.44
text-formatting language	שְׂפַת תִּסְדוּר, שְׂפַת תִּסְדוּר תַּמְלִיל	07.01.31
third-generation language	שְׂפַת דּוֹר שְׁלִישִׁי	07.01.13
thread	תַּהֲלִיכוֹן	07.10.16
three-address instruction	פְּקֻדַּת שְׁלֹשָׁה מְעָנִים	07.09.15
total correctness	נְכוֹנוּת כּוֹלֶלֶת	07.07.31
trace	עִקְבָה	07.06.07
trace [v]	עָקַב	07.06.08
trace program	תָּכְנִית (תוכנית) מַעֲקָב	07.07.16
translate [v]	תִּרְגֵּם	07.04.05
translation	תִּרְגּוּם	07.04.06
translation duration, translation time (2)	מֶשֶׁךְ תִּרְגּוּם	07.04.63
translation program	תָּכְנִית תִּרְגּוּם	07.04.07
translation time (1)	זְמַן תִּרְגּוּם (1)	07.04.60
translator	מְתַרְגֵּם [שם עצם]	07.04.07
translator directive	הַנְחָיַת מְתַרְגֵּם	07.04.66
two-address instruction	פְּקֻדַּת שְׁנֵי מְעָנִים	07.09.14
unconditional jump instruction	פְּקֻדַּת קְפִיצָה לֹא-מֻתְנֵית	07.09.28

English Term	מונח עברי	מספר מונח
underflow exception	חֲרִיג חֲמִיקָה	07.06.55
understandability	מוּבָנוּת	07.12.03
unrecoverable error	תַּקָּלָה לְלֹא תַּקָּנָה	07.07.36
utility program	תָּכְנִית (תוֹכנית) שֵׁרוּת	07.11.12
utility routine	שִׁגְרַת שֵׁרוּת	07.11.13
variable trace	עִקְבַת מִשְׁתַּנִּים	07.06.14
virtual address	מַעַן וִירְטוּאָלִי	07.09.42
wall-clock time	זְמַן שָׁעוֹן	07.06.04
work area	שֶׁטַח עֲבוֹדָה	07.02.15
work space	שֶׁטַח עֲבוֹדָה	07.02.15
working area	שֶׁטַח עֲבוֹדָה	07.02.15
working space	שֶׁטַח עֲבוֹדָה	07.02.15
zero-address instruction	פְּקֵדָה לְלֹא מַעַן	07.09.12