טופס הצעת פרויקט

י"ג הנדסת תוכנה

בן דוד איבגי

‏2022

תוכן עניינים

[פרטי הסטודנט והמוסד 2](#_Toc121126760)

[רקע תיאורטי בתחום הפרויקט 3](#_Toc121126761)

[תיאור הפרויקט 4](#_Toc121126762)

[הגדרת הבעיה האלגוריתמית 5](#_Toc121126763)

[תהליכים עיקריים בפרויקט 5](#_Toc121126764)

[שפות תכנות 5](#_Toc121126765)

[סביבת עבודה 5](#_Toc121126766)

[לוחות זמנים 6](#_Toc121126767)

[נספחים – חתימות ואישורים 6](#_Toc121126768)

# פרטי הסטודנט והמוסד

**שם המוסד:** מכללת בארי WAN TEC

**סמל המוסד:**

**שם הסטודנט:** בן דוד איבגי

**ת.ז. הסטודנט:** 326064490

**שם הפרויקט:** פיתוח קומפיילר לשפת תכנות מומצאת

# רקע תיאורטי בתחום הפרויקט

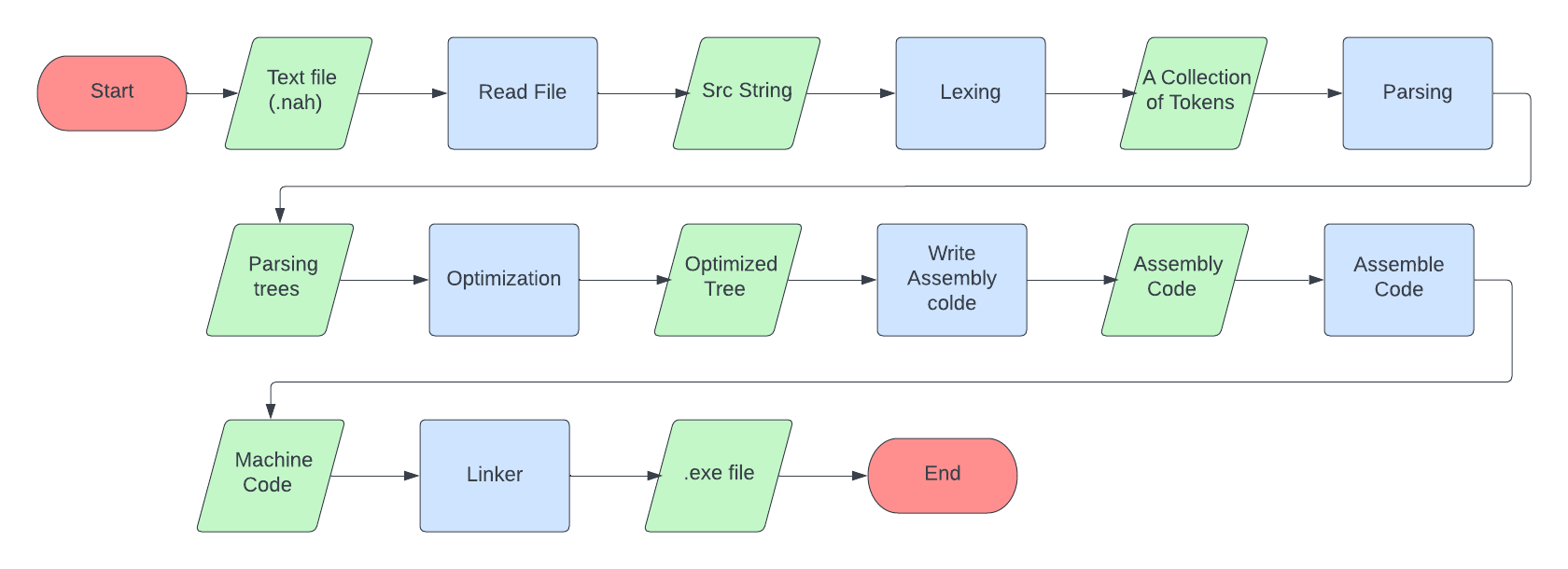
ככל שתחום התכנות התקדם ונעשה מסובך יותר, עלה הצורך בשפה שתהיה מובנת יותר לבני אדם, שתאפשר נוחות והבנה תיאורטית לצד האפשרות לחקר רעיונות מסובכים. הפתרון לבעיה זו היה פיתוח השפות העיליות במחצית השנייה של המאה הקודמת. שפות כמו COBOL, LISP, וALGOL הן רק חלק מהשפות שפותחו באותה התקופה, ועל אף הזיכרון המוגבל והקושי הרב שהיה כרוך בתכנות בשפות עיליות באותו זמן, הן השאירו חותם על תחום זה. מבנה השפות הללו היה דומה יותר לאנגלית מאשר לשפת מכונה – הוא הורכב מאוסף תווים (אלפבית), שבתורו היווה בסיס לבניית מילים, ואוסף המילים השמורות (יחד עם הכללים הדקדוקיים) היווה בתורו שפה, כמו כל שפה אחרת.

אולם ההתקדמות העלתה בעיה – כיצד ניתן לכתוב קוד בשפה שאינה מורכבת מאפסים ואחדים? הרי המחשב אינו דובר אנגלית, ואין לו דרך להבין למה אנחנו מתכוונים. היחידה המקשרת בין השפה לבין המחשב היא ה**מהדר**. המהדר (קומפיילר) אחראי על ההליך שבסופו מופק קוד קריא למחשב שלנו, והוא זה שמאפשר לנו לתכנת בשפות שדומות יותר לאנגלית מאשר לשפת מכונה. המהדרים הראשונים, שנכתבו בשנות החמישים, נכתבו ישירות בשפת מכונה, ולאחר מכן כללו גם שימוש בשפת אסמבלי (שפת ביניים). כיום, הרוב המוחלט של התוכנות כתוב בשפה עילית או בשפת ביניים, ועקב ההתפתחות בתחום החומרה, הבעיות שהאפילו על השימוש בשפה עילית הפכו ללא רלוונטיות.

המהדר מורכב מכמה תתי יחידות-

ראשית, הקוד מחולק לאסימונים (טוקניזציה), שהם יחידות בעלות ערך וסיווג (לדוגמה – {"שלום", מילה}, או {"15", מספר}). לאחר מכן כל משפט מנותח בצורה לוגית (פארסינג או ניתוח מחרוזת), והמשפט מורכב מחדש בהתאם למבנה הלוגי ולא למבנה הלינארי שלו (לדוגמה, עץ פארסינג). אחרי שמבוצע השלב הזה הקוד עובר ניתוח סמנטי (חיפוש אחר שגיאות, זיהוי משתנים בקוד), אופטימיזציה – הסרה של קוד חסר משמעות שאינו משפיע על מהלך התכנית, תרגום לשפת ביניים – לרוב שפת אסמבלי שמקלה על הביצוע כיוון שהיא מיתרגמת ישירות לשפת מכונה, ותרגום הקוד לשפת מכונה בעצמו. נוסף על המהדר קיים גם המקשר (לינקר), שאחראי ליצור את הקבצים הדרושים להרצת הקוד.

בתור מתכנת, הרגשתי שמדובר בנושא שעליי "לתקוף" כדי להבין לעומק את תחום התכנות. לאחר מחקר וניתוח של התהליכים שהקוד עובר עד שהוא מגיע לכדי ביצוע המסקנה הייתה חד משמעית – מדובר בתחום שיש לי זיקה אליו, ושאני מרגיש שיכול לתרום להתפתחות שלי מבחינה לימודית ומעשית כאחד.

(בתמונה – תרשים המתאר את התהליכים המתרחשים במהדר).

# תיאור הפרויקט

הפרויקט עצמו מחולק ליחידות שונות בהתאם לשלבי העבודה של המהדר:

- עיבוד הקלט (קריאה מקובץ) מעביר את הטקסט מהקובץ אל התוכנה.

- הלקסר מחלק את הטקסט לאסימונים.

- הפארסר ממיר את המשפטים הליניאריים לעץ בו קיים יחס לוגי בין המרכיבים אותו.

- הניתוח הסמנטי מחפש שגיאות בכתיבה ומוודא שמשתנים הוגדרו טרם השימוש בהם.

- האופטימיזציה מסירה קוד לא רלוונטי, מציבה קוד של פונקציה במקום בו היא נקראה, מחשבת ערכים קבועים, ובכך מייעלת את הקוד.

- התרגום לשפת הביניים ימיר את הביטויים הלוגיים המיועלים לשפת אסמבלי.

(תהליך ההמרה לשפת מכונה יתבצע ע"י DOSBOX)

- הכתיבה למסך (קונסול) תציג הדפסות ותאפשר קליטת ערכים מהמשתמש.

נוסף על המהדר, השפה דורשת גם קבצים בסיסיים משל עצמה. כמה דוגמאות אפשריות:

-ספריית I/O הדפסה וקליטה.

- ספרייה מתמטית.

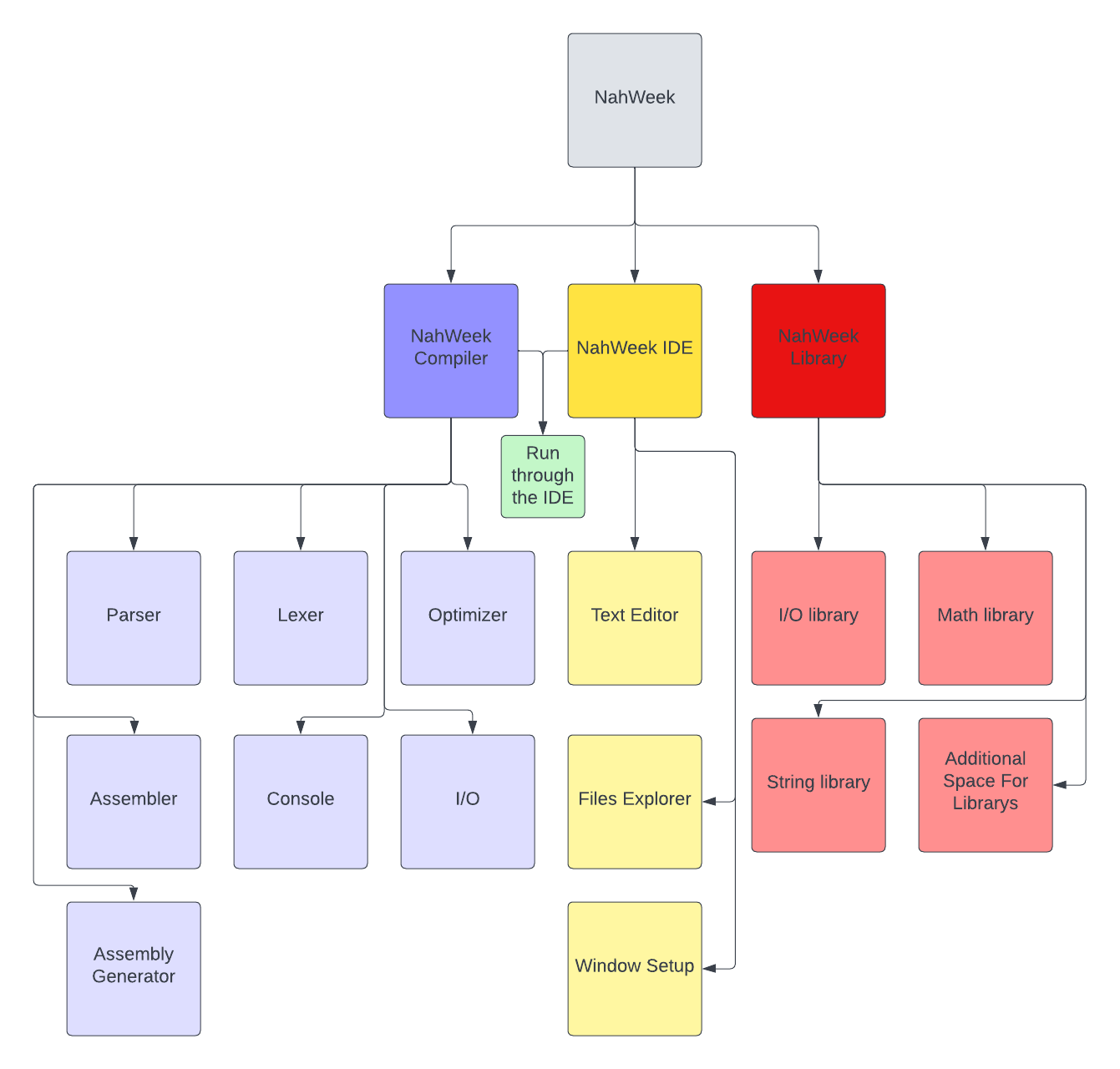
- ספרייה לטיפול במחרוזות.

במידה והדבר יתאפשר בהתאם ללוח הזמנים, ארצה לבנות גם עורך טקסט לשפה. העורך יורכב מ:

- מערכת לניהול קבצים (פתיחת קובץ, שמירת קובץ, חיפוש קובץ).

- כתיבה לקובץ ועריכת טקסט.

- הרצה דרך העורך (כפתור שיפעיל את הקומפיילר ויציג CONSOLE).



# הגדרת הבעיה האלגוריתמית

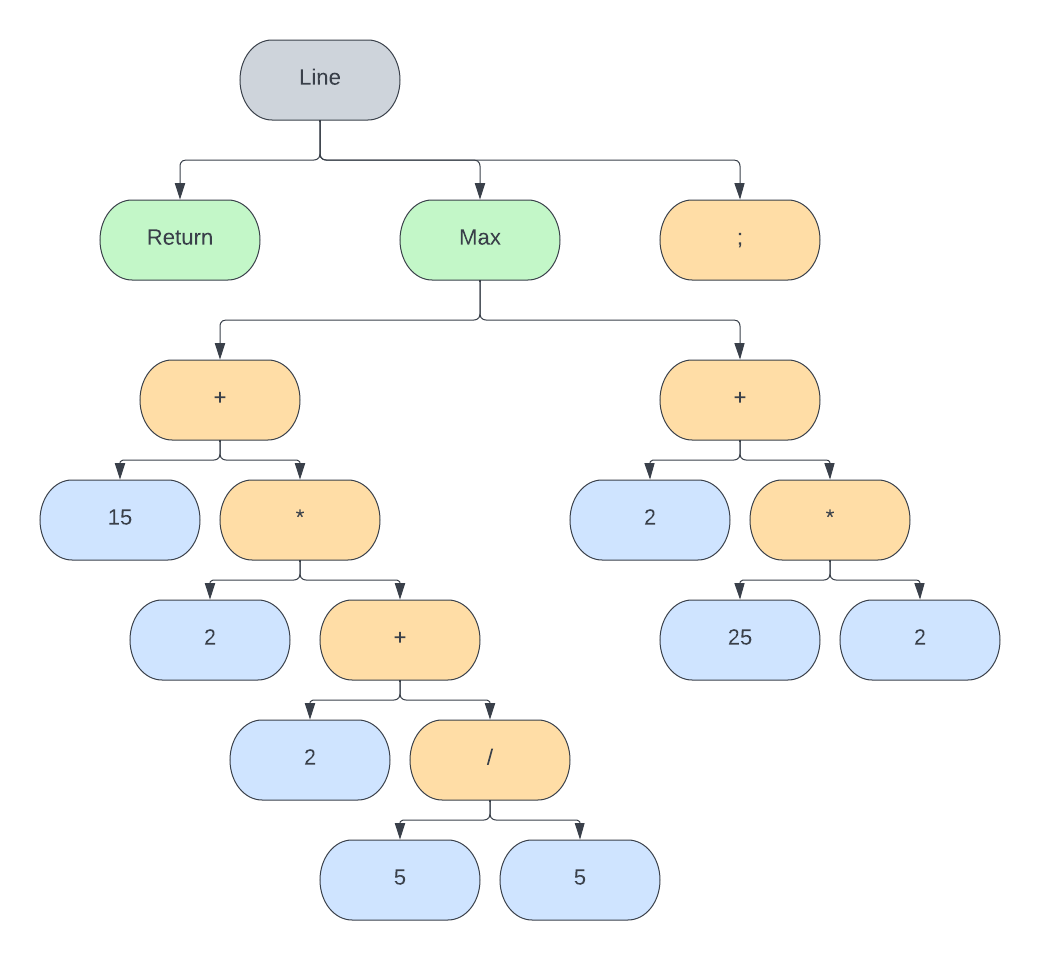
הבעיה האלגוריתמית היא בעיה פשוטה להבנה, אך מורכבת לפתרון – כיצד מתרגמים שפת תכנות לשפת מכונה? כיצד מונעים ומטפלים בכפל משמעויות, שגיאות בקוד שהמשתמש הזין, ובאותו הזמן מאפשרים תכנות שלא יעיק על המשתמש ויאפשר לו לתכנת בצורה שמביאה לידי ביטוי את החשיבה האלגוריתמית ויכולות התכנות שלו, ללא התעסקות מיותרת בבעיות דקדוקיות שונות.

במילים אחרות, יש להבין כיצד יתבצע כל אחד מהתהליכים השונים במהדר ביעילות מירבית (שפה שאינה בנויה כולה ממערכת של IF A -> DO B), וכיצד התהליכים מתקשרים אחד עם השני.

בתהליך הפארסינג, עלינו להציג היררכיה בין הגורמים בכל משפט, תוך חלוקה לסוגים שונים. במקרה כזה פתרון אפשרי הוא השימוש בעץ, בו כל גורם מתחלק לתתי הגורמים שלו.

**לדוגמה:**

הביטוי הלוגי **return max(15+2\*(5/5+2), 25\*2+2);** יוצג בצורה הבאה:



כאשר **[ס]** משמעותו מילת מפתח בשפה, **[ס]** הוא אופרטור ו- **[ס]** הוא אופרנד (ערך).

העץ מציג לנו תמונת מצב רחבה יותר על תתי האלמנטים בתוך הביטוי, ולכן נשתמש בו.

את העץ נבנה באמצעות פעולה רקורסיבית שתעבור על כל אלמנט במשפט ותבצע פעולות בהתאם לסוג האיבר.

**בניית העץ תתבצע בצורה הבאה:**

א. נעביר את המשפט לפורמט בו כל ביטוי מוקף בסוגריים.

כעת המשפט לדוגמה ייראה כך:

**return max ((15+(2\*((5/5) +2))), ((25\*2) +2))**

ב. בעבור כל מילה שטרם ביקרנו:

אם המילה היא מילת מפתח:

צור את המילה בעץ.

אם המילה מקבלת ערכים צור בן לכל ערך ו**קרא לפונקציה זו**

בעבור כל ערך של המילה.

אם המילה היא סוגריים פותחים:

צור בן שמאלי **וקרא לפונקציה זו** בעבור הבא.

אם המילה היא סוגריים סוגרים:

חזור להורה.

אם המילה היא אופרטור:

הכנס אל הצומת את האופרטור, צור בן ימני **וקרא לפונקציה זו** בעבור הבא.

אם המילה היא ערך:

הכנס אל הצומת את המספר וחזור להורה.

בצורה זו ניתן, באמצעות הרקורסיה, לבנות את עץ הפארסינג ולהקל בצורה משמעותית על המרת הקוד לשפת הביניים.

# תהליכים עיקריים בפרויקט

- עיבוד טקסט.

- המרה ל"אסימונים".

- המרה לביטויים לוגים.

- ניתוח סמנטי.

- אופטימיזציה של הקוד.

- המרה לשפת ביניים.

- המרה לשפת מכונה.

- המרה לקבצים המתאימים (.OBJ, .EXE)

- ממשק משתמש לכתיבת קוד.

- אפשרות להרצה דרך ממשק משתמש.

- תמיכה בכתיבה למסך (CONSOLE), וקליטה מהמסך.

- תמיכה בייבוא קבצים חיצוניים.

- תמיכה בכתיבת פונקציות, הגדרת משתנים.

- הרחבות אופציונליות : אובייקטים/ מבנים, ספריות מובנות (UTILS)

# שפות תכנות

תהליך ההמרה מטקסט לשפת הביניים, כמו גם ממשק המשתמש ומסך ההדפסות ייכתבו כולם בשפת C. השפה תיכתב בשפת Assembly (שפת ביניים), ספריות התמיכה ייכתבו בשפה המומצאת, וההמרה משפת ביניים לקובץ ההרצה יבוצע באופן אוטומטי ע"י DOSBOX (הפרויקט לא יכיל בנייה של אסמבלר).

# סביבת עבודה

Visual Studio, Notepad++, DOSBOX.

# לוחות זמנים

|  |  |
| --- | --- |
| שלב | תאריך |
| הגשת הצעת פרויקט | 14.12.2022 |
| שלב המחקר | 30.12.2022 |
| תכנון פרויקט (מבנה כללי) + בניית השפה (בחירת מילות מפתח + כללי דקדוק) | 15.01.2023 |
| בניית ההמרה מטקסט ל"אסימונים" | 25.01.2023 |
| בניית ההמרה לביטויים לוגיים | 05.02.2023 |
| בניית ההמרה לשפת ביניים | 20.02.2023 |
| בניית המסך לתקשורת עם המתכנת | 05.03.2023 |
| תמיכה בתכונות שונות (כמתואר בפרק "תהליכים עיקריים") | 20.03.2023 |
| בניית עורך הטקסט | 05.04.2023 |
| הוספת תכונות נוספות (כמתואר בפרק "תהליכים עיקריים: הרחבות") | תלוי הספק. |
| הגשת ספר פרויקט | 01.05.2023 |
| היבחנות | 15.05.2023 |

לוח הזמנים הוא דינאמי ובנוי רק על הערכות גסות, ייתכנו שינויים בהתאם לקצב ההתקדמות.

# נספחים – חתימות ואישורים

-חתימת הסטודנט

-חתימת רכזת המגמה

-אישור משרד החינוך