אפיון פרויקט מעבדה

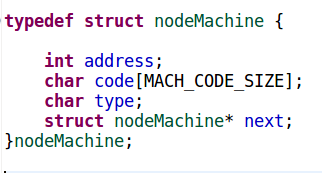
הפרוייקט נעשה ע"י שימי בן כליפה ת.ז 312438120 ובינמין .. תז

קבצי הפרוייקט:

**Main.c:**

הקובץ העיקר בפרוייקט , מהMAIN יופעלו האלגוריתמים המפורטים במטלה – קליטת קבצי הINPUT בדיקת תקינות , מעבר ראשון , מעבר שני ואם יש צורך יצירת הקבצים הנדרשים.

**Data.h:**

רשימה מקושרת חד כיוונית המחזיקה את הקוד הבינארי של כל פקודת אסמבלי מהקובץ הבנוי בצורה הבאה :

address – הכתובת של הפקודה

code – מערך המחזיק את הקוד

type – סוג הפקודה – האם הנחיה או הוראה

next – מצביע לאיבר הבא

* **void** **addMachineNode**(nodeMachine\*\* ptr\_head, **int** memValue, **char**\* machineCode,**char** type);

הפונקציה מוסיפה איבר לרשימה – במידה מדובר בפקודת הנחיה היא תתווסף בסוף הרשימה במידה ומדובר בפקודה הוראה היא תתווסף אחרי פקודת ההוראה האחרונה .

* **int** **checkIfMissing**(nodeMachine\*\* ptr, **int** Address);

הפונקציה בודקת האם לפקודה מסויימת יש 32 ביטים ( כלומר האם טופלה במלואה במעבר הראשון )

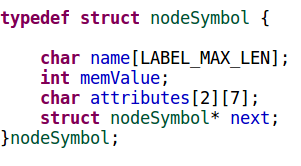
* **char**\* **getString**(nodeMachine\*\* ptr, **int** Address);

הפונקציה מקבל כתובת ומחזירה את הקוד הבינארי של הפקודה.

* void InitMachCode(char mach[]);

הפונקציה מאתחלת את מערך שמחזיק את הקוד הבינארי במהלך ניתוח הפקודות.

**Symbols.h:**

רשימה חד כיוונית המחזיקה את הסמלים של קובץ האסמבלי.

name – מערך המחזיק את שם הסמל

memValue – הכתובת של הסמל

attributes – מאפייני הסמל

next – מצביע לאיבר הבא

* **void** **addSymbolNode**(nodeSymbol\*\* p\_head, **char**\* name, **int** memValue, **char**\* attribute);

הפונקציה מוסיפה את הסמל לתחילת הרשימה.

* **int** **searchSymbol**(nodeSymbol\*\* p\_head, **char**\* name);

הפונקציה מקבלת שם של סמל ובודקת האם הוא נמצא ברשימה.

* **int** **returnAddress**(nodeSymbol\*\* p\_head, **char**\* name);

הפונקציה מקבלת שם של סמל ומחזירה את הכתובת שלו.

* **void** **addAttribute**(nodeSymbol\*\* p\_head, **char**\* name, **char**\* attribute);

הפונקציה מוסיפה מאפיין לסמל .

**firstPass.h:**

המעבר הראשון על הקלט , הכנסת כל הסמלים לטבלת הסמלים , והקודים לטבלת הדאטא ( חלקם באופן חלקי ) , תחזוק מונה הנתונים ומונה ההוראות.

* **int** **firstPass**(nodeSymbol\*\* head, nodeMachine\*\* ptr, nodeExtern\*\* e\_node, **int**\* DC, **int**\* IC, FILE\* filename);
* **int** **getBits**(**char**\* name);

הפונקציה מקבל של של הנחיה ומחזירה את מספר הביטים שלו.

* **int** **isGuidance**(**char**\* name);

הפונקציה מקבל שם של פקודה ומחזירה האם היא פקודת הנחיה.

* **int** **getFunct**(**char**\* name);

הפונקציה מקבלת שם של פקודה ומחזירה את ערך הFUNCT שלה .

* **int** **checkLimits**(**int** num);

הפונקציה מקבלת מספר ובודקת האם הוא חורג מגבולותיו.

* **char** **getType**(**char**\* name);

הפונקציה מקבלת שם של פקודה ומחזירה את הסוג שלו . ( סוג R , I או J)

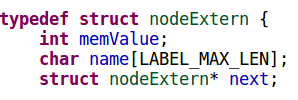
* **int** **getOpcode**(**char**\* name);

הפונקציה מקבלת שם של פקודה ומחזירה את הOPCODE שלו.

**secondPass.h:**

המעבר השני על הקלט , במעבר זה יושלמו הקודים שהיו חסרים אחרי המעבר הראשון כמו כן תתוחזק רשימה חד כיוונית המחזיקה את כתובות הסמלים החיצוניים.

* **int** **secondPass**(nodeSymbol\*\* head, nodeMachine\*\* ptr, nodeExtern\*\* e\_node, FILE\* filename);



memValue – הכתובת שבה הופיעה התווית

name – שם התווית

next – מצביע לאיבר הבא

* **void** **addNodeExtern**(nodeExtern\*\* p\_head, **char**\* name, **int** memValue);

הפונקציה מוסיפה איבר לטבלת הסמלים החיצוניים

* **int** **isExtern**(nodeSymbol\*\* p\_head, **char**\* name);

הפונקציה מקבל שם של תוית ומחזירה 1 אם התווית היא תווית חיצונית אחרת 0 .

* **int** **checkDuplAddressateExtern**(nodeSymbol\*\* p\_head, **char**\* name);

הפונקציה מקבלת שם של תווית פנימית ובודקת האם קיימת תווית חיצונית בעל אותו השם , מחזירה 1 במידה וכן , אחרת תחזיר 0.

**helpers.h:**

פונקציות עזר כלליות

* **char**\* **decimal\_to\_binary**(**int** n, **int** k);

פונקציה המקבלת מספר n ואורך k וממירה אותה למספר בינארי באורך k.

* **char**\* **convertToHex**(**char**\* binaryString);

הפונקציה מקבלת מצביע לסטרינג של קוד בינארי וממירה אותו להקסא.