**קבצי ראש- Headerfiles**

**מבוסס על מבנהstruct) ):** מבנה של בתים שבד”כ מכיל כמה שדות מידע, בד”כ יהיה בתחילת קובץ לפני מקטעי DATA .

יש להגדיר את המבנה כtypedef.

**שדות מידע רלוונטיים:**

* int version;
* int serialNum;
* int ItemsCount;
* char date[];
* char reserve[];

**File.h - קובץ סורס (סיומת h.):**

קובץ סורס שבתוכו יש הצהרת פונקציות, קבועים, מבנים ללא גוף פונקציה.

כאשר אנחנו נכתוב קובץ מקור-סורס עם פונקציות, אנחנו נייצר לצידו קובץ header בהתאמה.

לדוגמא אם נכתוב קובץ stack.c אז בהתאמה יהיה לידו קובץ .stack.h

**כדי שנוכל להשתמש בקובץ סורס עלינו לעשות include לדוגמא:**

#include "Week4\_3\_UseFunc.h"

**הקוד עצמו:**

#define FILE\_NAME "Data.bin"

#define Version 1

typedef struct {

    int version;

    int serialNum;

    int itemsCount;

    char reserve[100];

} t\_headerFile;

t\_headerFile fileHeader;

t\_item\* newItem(int key, int val)

{

    fileHeader.itemsCount++;

    t\_item\* ret= (t\_item\*)malloc(sizeof(t\_item));

    ret->key = key;

    ret->val = val;

    if (head == NULL)

    {

        head = ret;

        tail = ret;

        ret->next = NULL;

        ret->prev = NULL;

    }

    else

    {

        tail->next = ret;

        ret->prev = tail;

        ret->next = NULL;

        tail = ret;

    }

    return ret;

}

//save list and header in file

void saveListInFile(char fileName [20])

{

    t\_item\* curr = head;

    FILE\* f = fopen(fileName, "w");

    if (!f)

    {

        //error

        return 1;

    }

    // write the header into file

    fwrite(&fileHeader, sizeof(t\_headerFile), 1, f);

    //write all items into file

    while (curr != NULL)

    {

        fwrite(curr, sizeof(t\_item), 1, f);

        curr = curr->next;

    }

    fclose(f);

}

//build the list by reading from file

void buildListByFile(char fileName[20])

{

    t\_headerFile headerOfFile;

    //read from file

    FILE\* f = fopen(fileName, "r");

    if (!f)

    {

        //error

        return 1;

    }

    int read = fread(&headerOfFile, sizeof(t\_headerFile), 1, f);

    if (read == 0)

    {

        //error

        return 1;

    }

    //build the list

    head = NULL;

    tail = NULL;

    for (int i = 0; i < headerOfFile.itemsCount; i++)

    {

        t\_item\* curr = (t\_item\*)malloc(sizeof(t\_item));

        read = fread(curr, sizeof(t\_item), 1, f);

        newItem(curr->key, curr->val);

    }

    fclose(f);

**כאשר רוצים לבנות רשימה על פי בדיקת גרסה בקובץ ראש:**

// build the list

Head = NULL;

Tail = NULL;

if (headerOfFile.version == 1)

}

int id = 0;

// Convert Process

t\_Item\* curr;

for (int i = 0; i < headerOfFile.ItemsCount; i++)

{

curr = (t\_Item\*)malloc(sizeof(t\_Item));

read = fread(curr, sizeof(t\_Item), 1, f);

t\_Item2\* newI= NewItem2(curr->val, curr->key,id);

id++;

free(curr);

{

}

else if (headerOfFile.version == 2)

}

t\_Item2\* curr;

for (int i = 0; i < headerOfFile.ItemsCount; i++)

}

curr = (t\_Item2\*)malloc(sizeof(t\_Item2));

read = fread(curr, sizeof(t\_Item2), 1, f);

NewItem2(curr->val, curr->key,curr->id);

free(curr);

{

{

return 0;

{