**דוח מעבדה תרגיל בית 2**

**שמות מגישים:**

סער ויקטור – 312392822

אילון בן סימון – 312162951

**תרגיל 1**

קוד התכנית:

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <math.h>

#include <conio.h>

//decleration of structures

typedef struct Student

{

char \*Name;

long ID;

int GradeA;

int GradeB;

int HW;

}Student;

typedef struct University

{

int NumOfStudents;

Student \*array;

}University;

//decleration of funtions

void BuildArray(FILE\* in, University\* univ);

void ScanStudents(FILE\* in, Student\* stud);

void PrintArray(FILE\* out, University\* univ);

void FinalGrades(University\* univ, FILE\* out);

int CalcFinalG(float num, int HW);

void Stats(FILE\* out, University\* univ);

void BestGrade(FILE\* out, University\* univ);

void EndOfProgram(FILE\* out);

char UserChoice();

void Error\_Msg(char \*str);

FILE\* OpenFile(FileName, Mode);

//the main function shows a menu to the user and calls all the secondary functions

int main()

{

int i;

char user;

FILE \*in, \*out;

University Arr;

in = OpenFile("input.txt", "rt");

while (1)

{

user = UserChoice();

if (user == 'a')

{

BuildArray(in, &Arr);

fseek(in, 0, SEEK\_SET);

break;

}

else

printf("PLEASE ENTER 'a' FIRST!.\n");

}

user = UserChoice();

while (user!='f') //loop to operate all the functions

{

switch (user)

{

case 'a':

BuildArray(in, &Arr);

user = UserChoice();

fseek(in, 0, SEEK\_SET);

break;

case 'b':

out = OpenFile("output.txt", "wt");

PrintArray(out, &Arr);

fclose(out);

printf("File Has Been Updated!\n\n");

user = UserChoice();

break;

case 'c':

out = OpenFile("output.txt", "wt");

fprintf(out, "BEFORE:\n");

PrintArray(out, &Arr);

fprintf(out, "\n");

fprintf(out, "AFTER:\n");

FinalGrades(&Arr, out);

fclose(out);

printf("File Has Been Updated!\n\n");

user = UserChoice();

break;

case 'd':

out = OpenFile("output.txt", "wt");

Stats(out, &Arr);

fclose(out);

printf("File Has Been Updated!\n\n");

user = UserChoice();

break;

case 'e':

out = OpenFile("output.txt", "wt");

BestGrade(out, &Arr);

fclose(out);

printf("File Has Been Updated!\n\n");

user = UserChoice();

break;

default:

printf("INPUT ERROR!\n\n");

user = UserChoice();

}

} //end loop

out = OpenFile("output.txt", "wt");

EndOfProgram(out);

for (i = 0; i < Arr.NumOfStudents; i++) //freeing the name string of each student

free(Arr.array[i].Name);

free(Arr.array);

fclose(in);

fclose(out);

getch();

return 0;

}

//the function scans the information of one student

void ScanStudents(FILE\* in, Student\* stud)

{

char TempName[100];

fscanf(in, "%s %ld %d %d %d", TempName, &(stud->ID), &(stud->GradeA), &(stud->GradeB), &(stud->HW));

stud->Name = (char\*)malloc(strlen(TempName) + 1);

if (stud->Name == NULL)

Error\_Msg("Error! Memory Not Allocated");

strcpy(stud->Name, TempName);

}

//a.the function builds the array of Student\* structures

void BuildArray(FILE\* in, University\* univ)

{

int i = 1;

univ->array = (Student\*)malloc(sizeof(Student));

if (univ->array == NULL)

Error\_Msg("Error! Memory Not Allocated");

ScanStudents(in, &univ->array[i - 1]); //scans the information of the first student

while (!feof(in))

{

univ->array = (Student\*)realloc(univ->array, sizeof(Student)\*(i + 1));

if (univ->array == NULL)

Error\_Msg("Error! Memory Not Allocated");

ScanStudents(in, &(univ->array[i]));

i++;

}

univ->NumOfStudents = i;

}

//b.the function prints the entire array of students

void PrintArray(FILE\* out, University\* univ)

{

int i;

for (i = 0; i < univ->NumOfStudents; i++)

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].HW);

}

//c.the function calculates the grades after the consideration of the HW grades

void FinalGrades(University\* univ, FILE\* out)

{

int i;

float A, B;

int tempA, tempB;

for (i = 0; i < univ->NumOfStudents; i++)

{

A = (float)univ->array[i].GradeA;

B = (float)univ->array[i].GradeB;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

if ((A >= 0 && A <= 100) && (B >= 0 && B <= 100)) //check if both grades aren't '102'

{

if ((A >= 55 && A <= 100) && (B >= 55 && B <= 100)) //in case both grades needs to change

{

tempA = CalcFinalG(A, univ->array[i].HW);

tempB = CalcFinalG(B, univ->array[i].HW);

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, tempA, tempB, univ->array[i].HW);

}

else if (A >= 55 && A <= 100) //checks and calculates the first grade

{

tempA = CalcFinalG(A, univ->array[i].HW);

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, tempA, univ->array[i].GradeB, univ->array[i].HW);

}

else if (B >= 55 && B <= 100) //checks and calculates the second grade

{

tempB = CalcFinalG(B, univ->array[i].HW);

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, tempB, univ->array[i].HW);

}

else

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].HW);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

else if ((A >= 0 && A <= 100) && B == 102)

{

if (A >= 55 && A <= 100)

{

tempA = CalcFinalG(A, univ->array[i].HW);

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, tempA, univ->array[i].GradeB, univ->array[i].HW);

}

else

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].HW);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

else if ((B >= 0 && B <= 100) && A == 102)

{

if (B >= 55 && B <= 100)

{

tempB = CalcFinalG(B, univ->array[i].HW);

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, tempB, univ->array[i].HW);

}

else

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].HW);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

else

fprintf(out, "Student %d: %s %ld A: %d B: %d Hw: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].HW);

}

}

//the function checks if the final grade needs to be rounded up

int CalcFinalG(float num, int HW)

{

int tempNum;

num = num\*(0.8) + (HW)\*(0.2);

if ((int)(num + 1) == (int)(num + 0.5))

tempNum = (int)(num + 1);

else

tempNum = (int)num;

return tempNum;

}

//d.the function calculates the statistics of a specific moed

void Stats(FILE\* out, University\* univ)

{

char A\_Or\_B;

float avg, sum = 0, counter = 0, Sd;

int i, maxGrade = -1, minGrade = 101, NumOfStud = 0;

while (1)

{

printf("Enter which test you want to see the statistics of (A / B): ");

scanf(" %c", &A\_Or\_B);

if (A\_Or\_B == 'A' || A\_Or\_B == 'B' || A\_Or\_B == 'a' || A\_Or\_B == 'b')

break;

else

printf("Wrong Input!\n");

}

if (A\_Or\_B == 'A' || A\_Or\_B == 'a') //in case the user put moed A

{

for (i = 0; i < univ->NumOfStudents; i++)

{

if (univ->array[i].GradeA != 102)

{

sum += univ->array[i].GradeA;

NumOfStud++;

}

}

avg = sum / NumOfStud;

for (i = 0; i < univ->NumOfStudents; i++)

{

if (univ->array[i].GradeA != 102)

{

counter = counter + (univ->array[i].GradeA - avg)\*(univ->array[i].GradeA - avg);

}

}

Sd = sqrt(counter / NumOfStud);

for (i = 0; i < univ->NumOfStudents; i++)

{

if (univ->array[i].GradeA != 102)

{

if (maxGrade < univ->array[i].GradeA)

maxGrade = univ->array[i].GradeA;

if (minGrade > univ->array[i].GradeA)

minGrade = univ->array[i].GradeA;

}

}

}

else //in case the user put moed B

{

for (i = 0; i < univ->NumOfStudents; i++)

{

if (univ->array[i].GradeB != 102)

{

sum += univ->array[i].GradeB;

NumOfStud++;

}

}

avg = sum / NumOfStud;

for (i = 0; i < univ->NumOfStudents; i++)

{

if (univ->array[i].GradeB != 102)

{

counter = counter + (univ->array[i].GradeB - avg)\*(univ->array[i].GradeB - avg);

}

}

Sd = sqrt(counter / NumOfStud);

for (i = 0; i < univ->NumOfStudents; i++)

{

if (univ->array[i].GradeB != 102)

{

if (maxGrade < univ->array[i].GradeB)

maxGrade = univ->array[i].GradeB;

if (minGrade > univ->array[i].GradeB)

minGrade = univ->array[i].GradeB;

}

}

}

fprintf(out, "1.The average of all grades (moed %c): %.2f", A\_Or\_B, avg);

fprintf(out, "\n2.The standart deviation (moed %c): %.2f", A\_Or\_B, Sd);

fprintf(out, "\n3.Number of students attended (moed %c): %d", A\_Or\_B, NumOfStud);

fprintf(out, "\n4.The range of grades (moed %c): %d - %d", A\_Or\_B, maxGrade, minGrade);

}

//e.the function prints to the output file the best grade of the two

void BestGrade(FILE\* out, University\* univ)

{

int i;

for (i = 0; i < univ->NumOfStudents; i++)

{

if (univ->array[i].GradeA != 102 && univ->array[i].GradeB == 102)

{

fprintf(out, "Student %d: %s %ld A: %d B: %d Final: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].GradeA);

}

else if (univ->array[i].GradeA == 102 && univ->array[i].GradeB != 102)

{

fprintf(out, "Student %d: %s %ld A: %d B: %d Final: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].GradeB);

}

else if (univ->array[i].GradeA>univ->array[i].GradeB)

{

fprintf(out, "Student %d: %s %ld A: %d B: %d Final: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].GradeA);

}

else if (univ->array[i].GradeB>univ->array[i].GradeA)

{

fprintf(out, "Student %d: %s %ld A: %d B: %d Final: %d\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB, univ->array[i].GradeB);

}

else

{

fprintf(out, "Student %d: %s %ld A: %d B: %d Final: NOT GRADED!\n", i + 1, univ->array[i].Name, univ->array[i].ID, univ->array[i].GradeA, univ->array[i].GradeB);

}

}

}

//f.the function prints to the output file "End of program"

void EndOfProgram(FILE\* out)

{

fprintf(out, "End Of Program");

printf("Program has ended!");

}

//the function shows the menu to the user and scans his choice

char UserChoice()

{

char a;

printf("Enter your choice (first enter 'a'):\n");

printf("a. Scan the data of the students from the file.\n");

printf("b. Print the data of all students into the output file.\n");

printf("c. Calculate the final grade of each moed seperately.\n");

printf("d. Show the statistics of a moed to your choice.\n");

printf("e. Show the best grade of each moed.\n");

printf("f. End of program.\n");

scanf(" %c", &a);

system("cls"); //clear the output screen

return a;

}

//the function prints Error Msg and closes the program

void Error\_Msg(char \*str)

{

printf("\n%s", str);

exit(1);

}

//the function opens a file

FILE\* OpenFile(FileName, Mode)

{

FILE\* f;

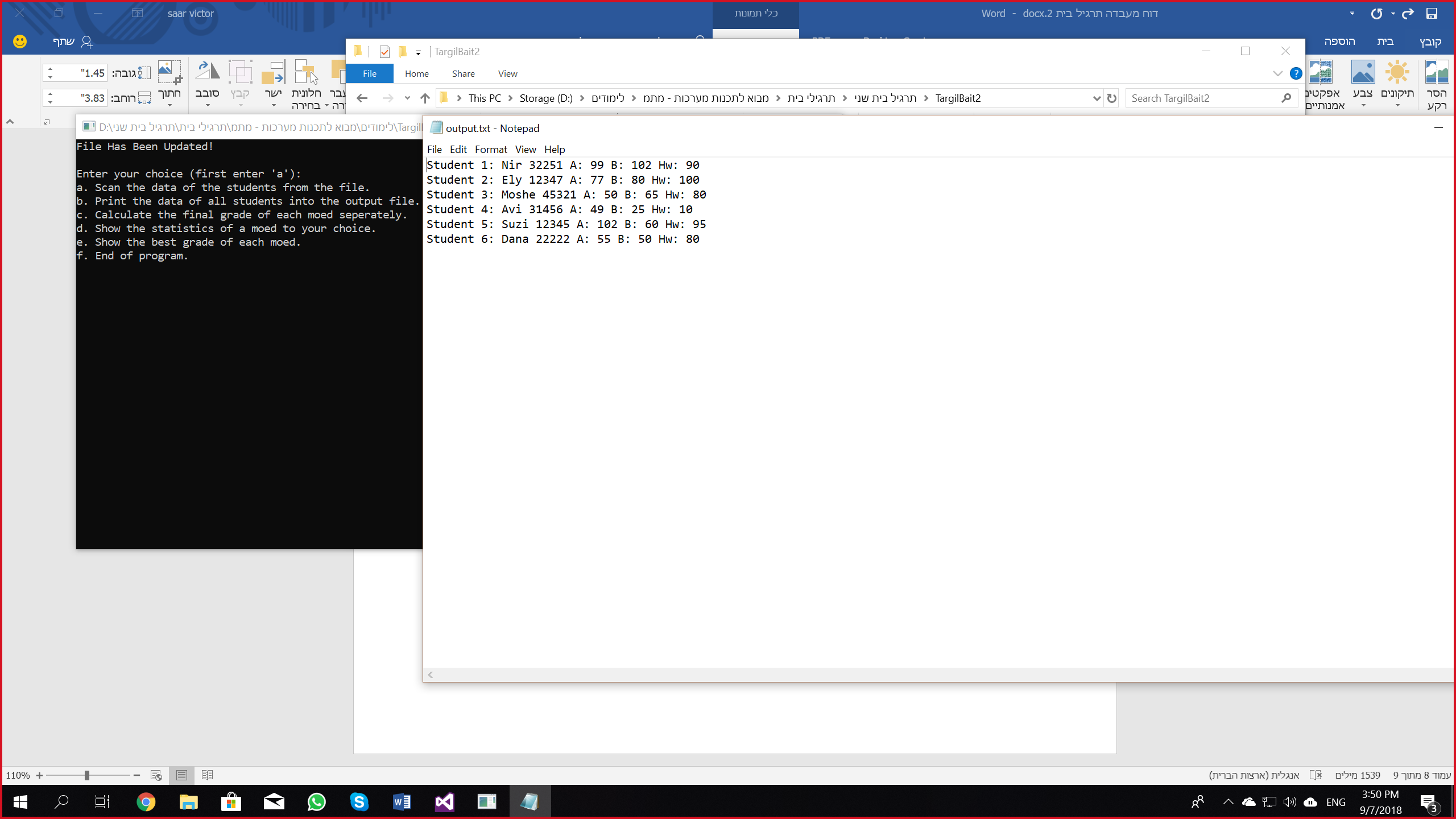
f = fopen(FileName, Mode);

if (f == NULL)

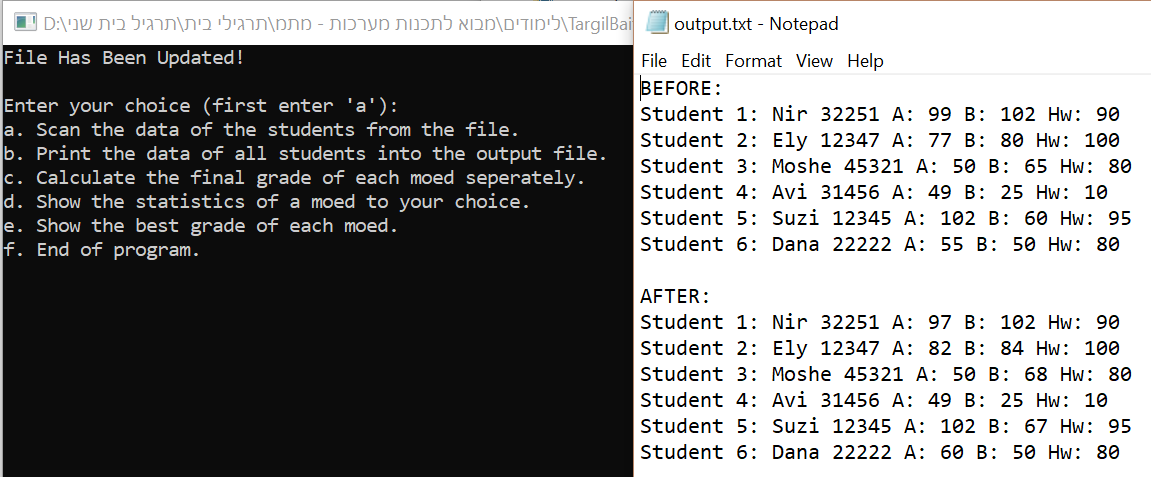
Error\_Msg("File Was Not Opened!");

return f;

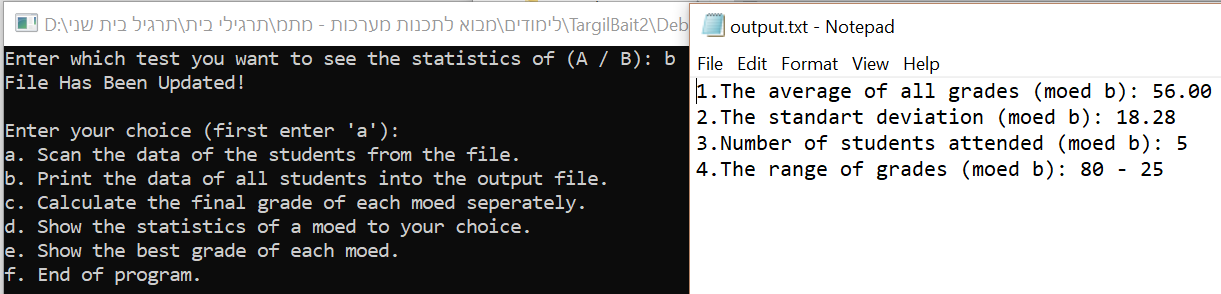
}

פלט 1 לדוגמא (מימוש פונקציה b):

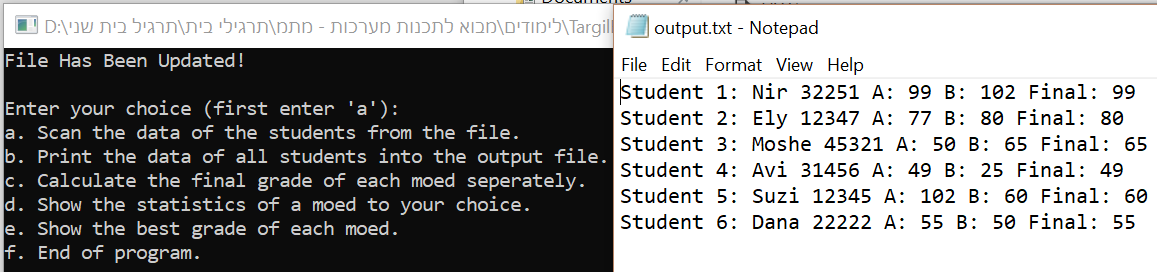
פלט 2 לדוגמא (מימוש פונקציה c):



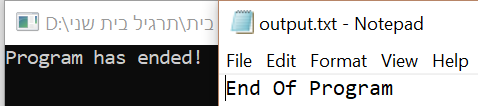
פלט 3 לדוגמא (מימוש פונקציה d):



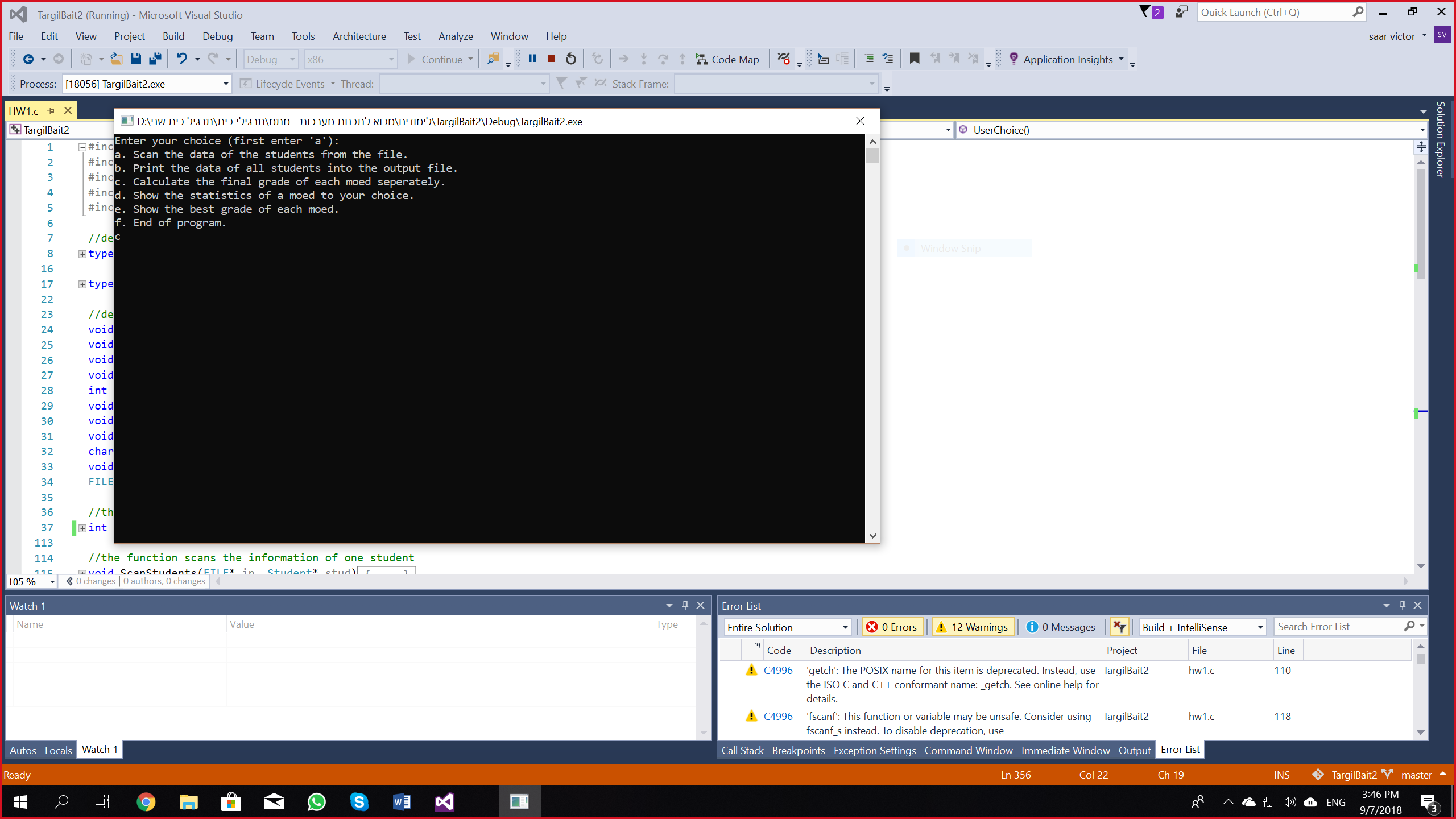
פלט 4 לדוגמא (מימוש פונקציה e):

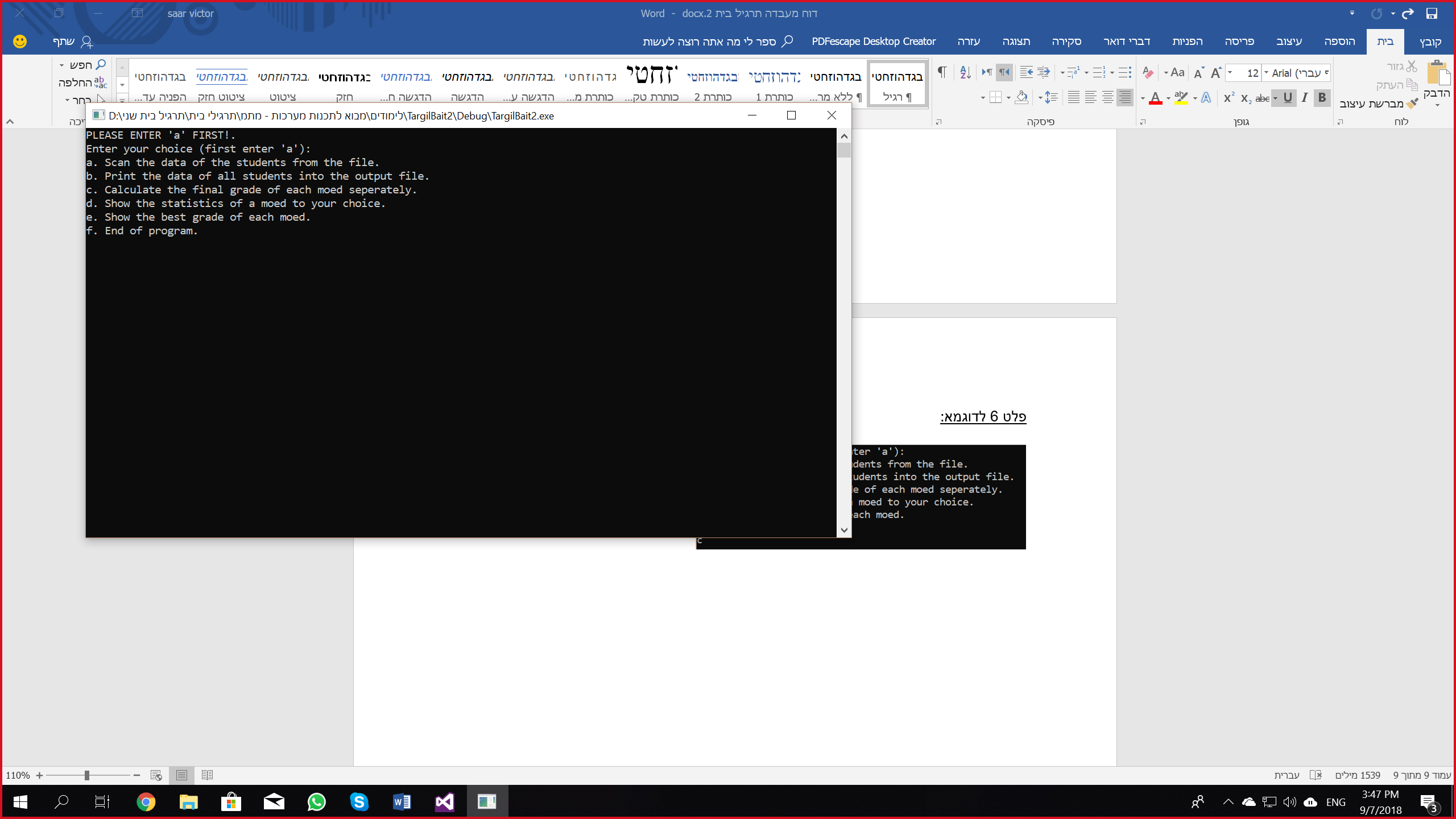


פלט 5 לדוגמא (מימוש פונקציה f):



פלט 6 לדוגמא:





פלט 7 לדוגמא:

