**מעבדה 10 – מת"מ -   
מגישים:  
אורי מלכא – 314862996  
אלן ציפין - 313206062  
  
שאלה 1 קוד  
פלט1:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

double sum\_square(int m, int n, float(\*f)(int a)); //function that receive 2 integers and a pointer to a function that return float and receives 1 integer

float func1(int x);

float func2(int x);

int main() {

printf("The sum of func2: %.5lf\n", sum\_square(2, 13, func2));

printf("The sum of func1: %.5lf\n", sum\_square(1, 10000, func1));

return 0;

}

float func1(int x) { return 1.0 / x; }

float func2(int x) { return x / 5.0; }

double sum\_square(int m, int n, float(\*f)(int a)) {

double sum = 0; //the function return double

float temp; //we use it to receive the float value from each function

int i;

for (i = m; i <= n; i++) {

temp = f(i); //we send a number between m<=i<=n

temp \*= temp; //f^2

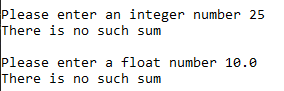
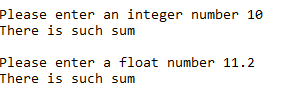
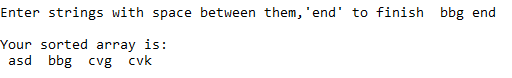
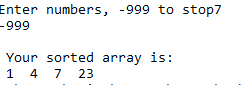
sum += temp;

}

return sum;

}



**שאלה 2 קוד:  
  
  
  
שאלה 2 פלט:** **שאלה 3 קוד:  
  
  
  
  
  
  
שאלה 3 פלטים:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<stdio.h>

#include<stdlib.h>

#define N 5

typedef enum { FALSE, TRUE } BOOL;

BOOL Int\_Sum(void\*, void\*, void\*);

BOOL Float\_Sum(void\*, void\*, void\*);

BOOL Sum(BOOL(\*f)(void\*, void\*, void\*), void\*\* p\_num, void\* number);//function the recive a pointer to a function that return ture/false and gets 3 void pointers. also recives a pointer to a pointers array and a pointer to a number.

int main()

{

int num[] = { 3,5,23,5,6 }, i, value;

float fnum[] = { 3.5,5.0,2.3,5.8,6.2 }, fvalue;

void\* p\_num[N], \* target;

printf("\nPlease enter an integer number ");

scanf("%d", &value);

target = &value; //we put the targted value into a void\* just to do more, we can also send the adress of &fvalue.

for (i = 0; i < N; i++) {

p\_num[i] = num+i;

}

if (Sum(Int\_Sum,p\_num,target) == TRUE)

printf("There is such sum\n");

else

printf("There is no such sum\n");

printf("\nPlease enter a float number ");

scanf("%f", &fvalue);

target = &fvalue; //we put the targted value into a void\* just to do more, we can also send the adress of &fvalue.

for (i = 0; i < N; i++) {

p\_num[i] = fnum + i;

}

if (Sum(Float\_Sum, p\_num, target) == TRUE)

printf("There is such sum\n");

else

printf("There is no such sum\n");

}

BOOL Int\_Sum(void\* a, void\* b, void\* c)

{

if (\*(int\*)a + \*(int\*)b == \*(int\*)c)

return TRUE;

return FALSE;

}

BOOL Float\_Sum(void\* a, void\* b, void\* c)

{

if (\*(float\*)a + \*(float\*)b == \*(float\*)c)

return TRUE;

return FALSE;

}

BOOL Sum(BOOL(\*f)(void\*, void\*, void\*), void\*\* p\_num, void\* number) {

int i, j;

for (i = 0; i < N - 1; i++) //the last element in the array is already checked with the whole array so its not necessary to check it.

for (j = i + 1; j < N; j++) //loop for b

{

if (f(p\_num[i], p\_num[j], number) == TRUE)

return TRUE;

}

return FALSE;

}

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#define N 5

typedef enum { FALSE, TRUE } BOOL;

void Insert(BOOL(\*f)(void\*, void\*), void\* Element, void\*\* Parray);

BOOL Int\_Comp(void\* FirstElement, void\* SecondElement);

BOOL IsFull();

int CurrentCount = 3; /\*global variable - only for the first ADT lab!Please,don't use!\*/

int main()

{

int Array[N] = { 1,4,23 }, Value, i;

void\* PArray[N]; /\*pointers array\*/

for (i = 0; i < N; i++) {

PArray[i] = Array + i;

}

printf("Enter numbers, -999 to stop");

scanf("%d", &Value);

while (Value != -999)

{

if (!IsFull())

{

Array[CurrentCount] = Value;

Insert(Int\_Comp, &Array[CurrentCount], PArray); /\*call the general function\*/

scanf("%d", &Value);

}

else

{

printf("The array is full");

break;

}

}

printf("\n Your sorted array is:\n"); /\*print the array in sorted order\*/

for (i = 0; i < CurrentCount; i++)

printf(" %d ", \*(int\*)PArray[i]);

return 0;

}

BOOL IsFull()

{

if (CurrentCount == N)

return TRUE;

return FALSE;

}

BOOL Int\_Comp(void\* FirstElement, void\* SecondElement)

{

if (\*(int\*)FirstElement > \* (int\*)SecondElement)

return TRUE;

return FALSE;

}

void Insert(BOOL(\*f)(void\*, void\*), void\* Element, void\*\* Parray) {

int i = 0, j;

while (f(Element, Parray[i]) == TRUE) i++; //once we finish the loop the i represent the index of the element that our element is greater then him. and the element need to be placed in the i+1 place.

for (j = CurrentCount; j >= i+1; j--) {

Parray[j] = Parray[j - 1];

}

Parray[i]= Element;

CurrentCount++; //we increase the size of the current array.

}