***Министерство образования Республики Беларусь***

***Учреждение Образования***

***«Брестский Государственный Технический Университет»***

***Кафедра ИИТ***

**Лабораторная работа №10**

**По дисциплине ОАиП за II семестр**

**Тема: «Функции»**

**Выполнил:**

Студент 1-го курса

Группы ИИ-15(1)

Волк И. А.

**Проверила:**

Хацкевич М. В.

Брест 2018

Цель: научиться работать с функциями.

Задание 1.

9. Описать процедуру RemoveForInc(A, N), удаляющую из вещественного массива A размера N «лишние» элементы так, чтобы оставшиеся элементы оказались упорядоченными по возрастанию: первый элемент не удаляется, второй элемент удаляется, если он меньше первого, третий — если он меньше предыдущего элемента, оставленного в массиве, и т. д.Например, массив 5.5, 2.5, 4.6, 7.2, 5.8, 9.4 должен быть преобразован к виду 5.5, 7.2, 9.4. Массив A и число N являются входными и выходными параметрами. С помощью этой процедуры преобразовать массивы A, B,С размера NA, NB, NC соответственно и вывести размер и содержимое полученных массивов.

Код программы:

#include "stdafx.h"

#include <Windows.h>

#include <iostream>

using namespace std;

const int notName = system("color f0");

int RemoveForInc(double\*, int);

void out(char\*, double\*, int);

int main()

{

setlocale(LC\_ALL, "rus");

double A[] = {1.0, 3.2, 2.0, 8.5, 1.2};

int sizeA = sizeof(A) / sizeof(\*A);

out("A", A, sizeA);

double B[] = { 4.3, 3.21, 2.58, 15.56, 56.5, 45.5 };

int sizeB = sizeof(B) / sizeof(\*B);

out("B", B, sizeB);

double C[] = { 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0 };

int sizeC = sizeof(C) / sizeof(\*C);

out("C", C, sizeC);

system("pause");

return 0;

}

int RemoveForInc(double \* A, int N)

{

int i = 1;

while (i < N)

{

if (A[i]<A[i - 1])

{

for (int j = i; j < (N - 1); j++)

A[j] = A[j + 1];

N--;

continue;

}

i++;

}

return N;

}

void out(char\* name, double\*A, int sizeA)

{

cout << name << " before: ";

for (int i = 0; i < sizeA; i++)

{

cout << A[i] << ((i == (sizeA - 1)) ? "." : ", ");

}

cout << endl << "Size of " << name << " after: " << sizeA << "." << endl;

cout << name << " after : ";

sizeA = RemoveForInc(A, sizeA);

for (int i = 0; i < sizeA; i++)

{

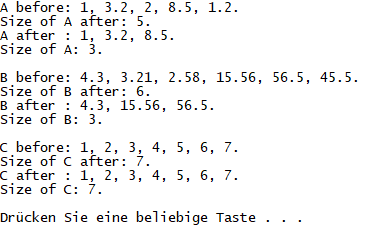
cout << A[i] << ((i<(sizeA - 1)) ? ", " : ".");

}

cout << endl << "Size of "<< name <<": " << sizeA << "." << endl << endl;

}

Output:



Задание 2.

11. Описать функцию PosK(S0, S, K) целого типа, возвращающую номер позиции, начиная с которой в строке S содержится K-е вхождение подстроки S0 (K > 0). Если количество вхождений S0 в строке S меньше K, то функция возвращает 0. Считать, что перекрывающихся вхождений подстрок S0 строка S не содержит. Вывести значения этой функции для пяти данных троек: S0, S и K.

Код программы:

#include "stdafx.h"

#include <Windows.h>

#include <iostream>

#include <stdio.h>

#include <string>

using namespace std;

const int notName = system("color f0");

char \* WordK(char\*, int);

int main()

{

char S[1024];

cout << " Enter the sentence: ";

gets\_s(S);

char \*\* K = new char\*[3];

for (int i = 1; i < 4; i++)

{

int numOfWord;

cout << "\nEnter the number of word: "; cin >> numOfWord;

K[i-1] = WordK(S, numOfWord);

cout << endl << " Word: ";

int j = 0;

while (K[i-1][j] != '\0')

{

cout << K[i-1][j]; j++;

}

cout << endl;

}

system("pause");

return 0;

}

char \* WordK(char \* S, int K)

{

int wordN = 0, i = 0, iW = 0;

char \* word = new char[1024];

while (1)

{

char \* chPointer = word + iW;

if (!iW)

{

if (S[i] == ' ') { i++; continue; }

else if (S[i] == '\0') return "";

else { \*chPointer = S[i]; i++; iW++; }

}

else

{

if (S[i] == ' ' || S[i] == '\0')

{

\*chPointer = '\0';

iW = 0;

wordN++;

if (wordN == K) return word;

else if (S[i] == '\0') return "";

i++;

}

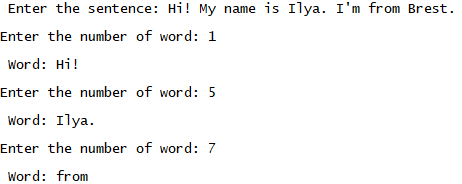
else { \*chPointer = S[i]; iW++; i++; }

}

}

}

Output:



Задание 3:

5. Описать рекурсивную функцию Fib2(N) целого типа, вычисляющую N-й элемент последовательности чисел Фибоначчи (N — целое число):

F1 = F2 = 1, FK = FK–2 + FK–1, K = 3, 4, … .

Считать, что номер N не превосходит 20. Для уменьшения количества рекурсивных вызовов по сравнению с функцией Fib1 (см. задание 4) создать вспомогательный массив для хранения уже вычисленных чисел Фибоначчи и обращаться к нему при выполнении функции Fib2. С помощью функции Fib2 найти пять чисел Фибоначчи с данными номерами.

Код программы:

#include "stdafx.h"

#include <Windows.h>

#include <iostream>

using namespace std;

const int notName = system("color f0");

int Fib2(int, int[]);

int main()

{

int \* arr = new int[20];

arr[0] = arr[1] = 1;

int \* res = new int[5];

for (int i = 0; i < 5; i++)

{

int num;

cout << "\n Number: "; cin >> num;

res[i] = Fib2(num, arr);

cout << " Fib2: " << res[i] << endl;

}

cout << endl;

system("pause");

return 0;

}

int Fib2(int N, int arr[])

{

if (N > 2)

{

Fib2(N - 1, arr);

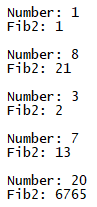
arr[N - 1] = arr[N - 2] + arr[N - 3];

}

return arr[N-1];

}

Output:



Вывод: по ходу лабораторной работы научился работать с функциями.

Блок-схема к заданию 1.

