УО «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра ПОИТ

Отчет по лабораторной работе №2.2

по предмету «Основы алгоритмизации и программирования»

Вариант 15

Выполнила:

Городко К. Е.

Гр. 351005

Проверила:

Данилова Г. В.

Минск 2023

**ЗАДАНИЕ:**

Два натуральных числа заданы последовательностями своих цифр.

Получить последовательность цифр, представляющую сумму данных

чисел (сложение столбиком).

**КОД ПРОГРАММЫ DELPHI:**

Program LAB2\_2;

Uses

System.SysUtils;

Const

MIN = 0;

MAX = 9;

MINNUM = 1;

MAXNUM = 1000000;

Type

TArrOI = Array Of Integer;

Procedure WriteCondition();

Begin

Writeln('Данная программа складывает два числа столбиком.');

Writeln('Условие ввода чисел: числа целые, от 1 до 1000000.');

End;

Function InputNum(Number: Integer): Integer;

Var

Num: Integer;

IsCorrect: Boolean;

Begin

Repeat

IsCorrect := True;

Write('Введите ', Number, ' число: ');

Try

Read(Num);

Except

Writeln('Введенные данные не соответствуют условию. Повторите попытку.');

IsCorrect := False;

End;

If IsCorrect And ((Num < MINNUM) Or (Num > MAXNUM)) Then

Begin

Writeln('Введенные данные не соответствуют условию. Повторите попытку.');

IsCorrect := False;

End;

Until IsCorrect;

InputNum := Num;

End;

Function FindNumSize(Num, Size: Integer): Integer;

Begin

Size := 0;

While Num > MIN Do

Begin

Inc(Size);

Num := Num Div 10;

End;

FindNumSize := Size;

End;

Function ConvertToArray(Num, Size: Integer): TArrOI;

Var

NumArr: TArrOI;

I, Ten: Integer;

Begin

SetLength(NumArr, Size);

Ten := 10;

For I := High(NumArr) DownTo 0 Do

Begin

NumArr[I] := (Num Mod Ten) Div (Ten Div 10);

Ten := Ten \* 10;

End;

ConvertToArray := NumArr;

End;

Function FindSum(BiggerSize, SmallerSize: Integer;

BiggerNum, SmallerNum: TArrOI): TArrOI;

Var

Sum: TArrOI;

I: Integer;

Begin

SetLength(Sum, BiggerSize);

For I := High(Sum) DownTo 0 Do

If (I > (BiggerSize - SmallerSize - 1)) Then

Sum[I] := BiggerNum[I] + SmallerNum[I - (BiggerSize - SmallerSize)]

Else

Sum[I] := BiggerNum[I];

FindSum := Sum;

End;

Function FindSumCondition(FstSize, SecSize: Integer;

FstNumArr, SecNumArr: TArrOI): TArrOI;

Var

Sum: TArrOI;

Begin

If (FstSize > SecSize) Then

Sum := FindSum(FstSize, SecSize, FstNumArr, SecNumArr)

Else

Sum := FindSum(SecSize, FstSize, SecNumArr, FstNumArr);

FindSumCondition := Sum;

End;

Procedure AddOnes(Var Sum: TArrOI);

Var

I, One: Integer;

Begin

One := 0;

For I := High(Sum) DownTo 1 Do

Begin

Sum[I] := Sum[I] + One;

If Sum[I] > MAX Then

Begin

One := 1;

Sum[I] := Sum[I] - 10;

End

Else

One := 0;

End;

Sum[0] := Sum[0] + One;

End;

Procedure AddPosition(Var Sum: TArrOI);

Var

I: Integer;

Begin

If Sum[0] > MAX Then

Begin

SetLength(Sum, (High(Sum) + 2));

For I := (High(Sum) - 1) Downto 0 Do

Sum[I + 1] := Sum[I];

Sum[1] := Sum[0] - 10;

Sum[0] := 1;

End;

End;

Procedure OutputArray(Arr, Sum: TArrOI);

Var

I: Integer;

Begin

For I := Low(Arr) To (High(Sum) - High(Arr)) Do

Write(' ');

For I := Low(Arr) To High(Arr) Do

Write(Arr[I]);

Writeln;

End;

Procedure OutputSum(FstNumArr, SecNumArr, Sum: TArrOI);

Var

I: Integer;

Begin

Writeln('Результат:');

OutputArray(FstNumArr, Sum);

Writeln('+');

OutputArray(SecNumArr, Sum);

Writeln('-------');

OutputArray(Sum, Sum);

End;

Var

FstNum, SecNum, FstSize, SecSize: Integer;

FstNumArr, SecNumArr, Sum: TArrOI;

Begin

WriteCondition();

FstNum := InputNum(1);

SecNum := InputNum(2);

FstSize := FindNumSize(FstNum, FstSize);

SecSize := FindNumSize(SecNum, SecSize);

FstNumArr := ConvertToArray(FstNum, FstSize);

SecNumArr := ConvertToArray(SecNum, SecSize);

Sum := FindSumCondition(FstSize, SecSize, FstNumArr, SecNumArr);

AddOnes(Sum);

AddPosition(Sum);

OutputSum(FstNumArr, SecNumArr, Sum);

End.

**КОД ПРОГРАММЫ С++:**

#include <iostream>

using namespace std;

const int MIN = 0;

const int MAX = 9;

const int MINNUM = 1;

const int MAXNUM = 1000000;

void writeCondition() {

cout << "Данная программа складывает два числа столбиком." << endl;

cout << "Условие ввода чисел: числа целые, от 1 до 1000000." << endl;

}

int inputNum(int number) {

int num = 0;

bool isIncorrect;

do {

isIncorrect = false;

cout << "Введите " << number << " число: ";

cin >> num;

if (cin.get() != '\n') {

cin.clear();

while (cin.get() != '\n');

isIncorrect = true;

cout << "Введенные данные не соответствуют условию. Повторите попытку."

<< endl;

}

if (!isIncorrect && ((num < MINNUM) || (num > MAXNUM))) {

isIncorrect = true;

cout << "Введенные данные не соответствуют условию. Повторите попытку."

<< endl;

}

} while (isIncorrect);

return num;

}

int findNumSize(int num) {

int size = 0;

while (num > MIN) {

size++;

num /= 10;

}

return size;

}

int\* convertToArray(int num, int size) {

int ten = 10;

int\* numArr = new int[size];

for (int i = size - 1; i > -1; i--) {

numArr[i] = ((num % ten) / (ten / 10));

ten \*= 10;

}

return numArr;

}

int findSumSize(int fstSize, int secSize) {

int sumSize = 0;

if (fstSize > secSize)

return fstSize;

else

return secSize;

}

int\* findSum(int biggerSize, int smallerSize, int sumSize, int\* biggerNum,

int\* smallerNum) {

int\* sum = new int[sumSize];

for (int i = (sumSize - 1); i > -1; i--)

if (i > (biggerSize - smallerSize - 1))

sum[i] = biggerNum[i] + smallerNum[i - (biggerSize - smallerSize)];

else

sum[i] = biggerNum[i];

return sum;

}

int\* findSumCondition(int fstSize, int secSize, int sumSize, int\* fstNumArr,

int\* secNumArr) {

if (fstSize > secSize)

return findSum(fstSize, secSize, sumSize, fstNumArr, secNumArr);

else

return findSum(secSize, fstSize, sumSize, secNumArr, fstNumArr);

}

int\* addOnes(int sumSize, int\* sum) {

int one = 0;

for (int i = (sumSize - 1); i > 0; i--) {

sum[i] += one;

if (sum[i] > MAX) {

one = 1;

sum[i] -= 10;

}

else

one = 0;

}

sum[0] += one;

return sum;

}

int\* addPosition(int sumSize, int\* sum) {

if (sum[0] > MAX) {

int\* sumNew = new int[sumSize];

for (int i = (sumSize - 2); i > -1; i--)

sumNew[i + 1] = sum[i];

sumNew[1] = sum[0] - 10;

sumNew[0] = 1;

delete[] sum;

return sumNew;

}

else return sum;

}

void outputArray(int numSize, int sumSize, int\* arr) {

for (int i = 0; i < (sumSize - numSize + 1); i++)

cout << " ";

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

cout << arr[i];

cout << endl;

}

void outputSum(int fstSize, int secSize, int sumSize, int\* fstNumArr, int\* secNumArr,

int\* sum) {

cout << "Результат: " << endl;

outputArray(fstSize, sumSize, fstNumArr);

cout << "+" << endl;

outputArray(secSize, sumSize, secNumArr);

cout << "-------" << endl;

outputArray(sumSize, sumSize, sum);

}

int main() {

setlocale(LC\_ALL, "Russian");

int fstNum = 0;

int secNum = 0;

int fstSize = 0;

int secSize = 0;

int sumSize = 0;

writeCondition();

fstNum = inputNum(1);

secNum = inputNum(2);

fstSize = findNumSize(fstNum);

secSize = findNumSize(secNum);

int\* fstNumArr = convertToArray(fstNum, fstSize);

int\* secNumArr = convertToArray(secNum, secSize);

sumSize = findSumSize(fstSize, secSize);

int\* sum = findSumCondition(fstSize, secSize, sumSize, fstNumArr, secNumArr);

sum = addOnes(sumSize, sum);

if (sum[0] > MAX)

sumSize++;

sum = addPosition(sumSize, sum);

outputSum(fstSize, secSize, sumSize, fstNumArr, secNumArr, sum);

return 0;

}

**КОД ПРОГРАММЫ JAVA:**

import java.util.Scanner;

public class Main {

static final int MIN = 0;

static final int MAX = 9;

static final int MINNUM = 1;

static final int MAXNUM = 1000000;

private static void writeCondition() {

System.out.println("Данная программа складывает два числа столбиком.");

System.out.println("Условие ввода чисел: числа целые, от 1 до 1000000.");

}

private static int inputNum(int number, Scanner scan) {

int num = 0;

boolean isIncorrect;

do {

isIncorrect = false;

System.out.print("Введите " + number + " число: ");

try {

num = Integer.parseInt(scan.nextLine());

}

catch (NumberFormatException e) {

System.out.println("Введенные данные не соответствуют условию.

Повторите попытку.");

isIncorrect = true;

}

if (!isIncorrect & ((num < MINNUM) | (num > MAXNUM))) {

isIncorrect = true;

System.out.println("Введенные данные не соответствуют условию.

Повторите попытку.");

}

} while (isIncorrect);

return num;

}

private static int findNumSize(int num) {

int size = 0;

while (num > MIN) {

size++;

num /= 10;

}

return size;

}

private static int[] convertToArray(int num, int size) {

int ten = 10;

int[] numArr = new int[size];

for (int i = size - 1; i > -1; i--) {

numArr[i] = ((num % ten) / (ten / 10));

ten \*= 10;

}

return numArr;

}

private static int findSumSize(int fstSize, int secSize) {

int sumSize = 0;

if (fstSize > secSize)

return fstSize;

else

return secSize;

}

private static int[] findSum(int biggerSize, int smallerSize, int sumSize,

int[] biggerNum, int[] smallerNum) {

int[] sum = new int[sumSize];

for (int i = (sumSize - 1); i > -1; i--)

if (i > (biggerSize - smallerSize - 1))

sum[i] = biggerNum[i] + smallerNum[i - (biggerSize - smallerSize)];

else

sum[i] = biggerNum[i];

return sum;

}

private static int[] findSumCondition(int fstSize, int secSize, int sumSize,

int[] fstNumArr, int[] secNumArr) {

if (fstSize > secSize)

return findSum(fstSize, secSize, sumSize, fstNumArr, secNumArr);

else

return findSum(secSize, fstSize, sumSize, secNumArr, fstNumArr);

}

private static int[] addOnes(int sumSize, int[] sum) {

int one = 0;

for (int i = (sumSize - 1); i > 0; i--) {

sum[i] += one;

if (sum[i] > MAX) {

one = 1;

sum[i] -= 10;

}

else

one = 0;

}

sum[0] += one;

return sum;

}

private static int[] addPosition(int sumSize, int[] sum) {

if (sum[0] > MAX) {

int[] sumNew = new int[sumSize];

for (int i = (sumSize - 2); i > -1; i--)

sumNew[i + 1] = sum[i];

sumNew[1] = sum[0] - 10;

sumNew[0] = 1;

return sumNew;

}

else return sum;

}

private static void outputArray(int numSize, int sumSize, int[] arr) {

for (int i = 0; i < (sumSize - numSize + 1); i++)

System.out.print(" ");

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

System.out.print(arr[i]);

System.out.println();

}

private static void outputSum(int fstSize, int secSize, int sumSize,

int[] fstNumArr, int[] secNumArr, int[] sum) {

System.out.println("Результат: ");

outputArray(fstSize, sumSize, fstNumArr);

System.out.println("+");

outputArray(secSize, sumSize, secNumArr);

System.out.println("-------");

outputArray(sumSize, sumSize, sum);

}

public static void main(String[] args) {

final Scanner scan = new Scanner(System.in);

int fstNum = 0;

int secNum = 0;

int fstSize = 0;

int secSize = 0;

int sumSize = 0;

writeCondition();

fstNum = inputNum(1, scan);

secNum = inputNum(2, scan);

scan.close();

fstSize = findNumSize(fstNum);

secSize = findNumSize(secNum);

int[] fstNumArr = convertToArray(fstNum, fstSize);

int[] secNumArr = convertToArray(secNum, secSize);

sumSize = findSumSize(fstSize, secSize);

int[] sum = findSumCondition(fstSize, secSize, sumSize, fstNumArr,

secNumArr);

sum = addOnes(sumSize, sum);

if (sum[0] > MAX)

sumSize++;

sum = addPosition(sumSize, sum);

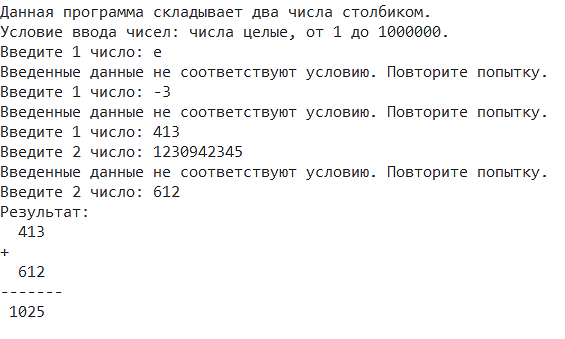
outputSum(fstSize, secSize, sumSize, fstNumArr, secNumArr, sum);

}

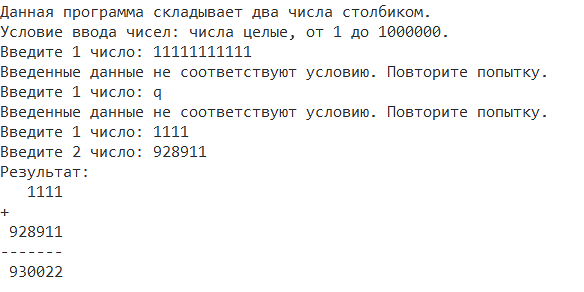
}

**СКРИНШОТЫ:**

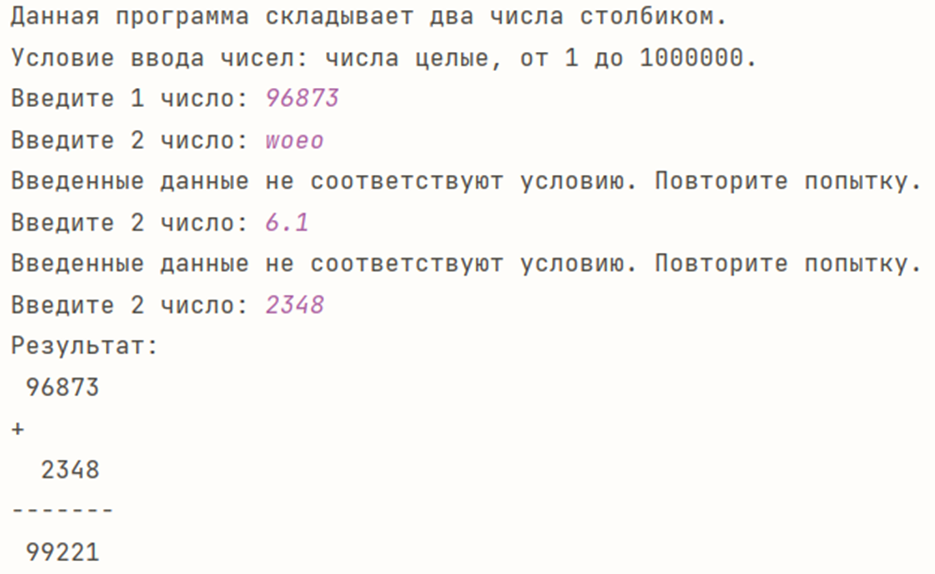
**Delphi:**



**C++:**



**Java:**



**БЛОК-СХЕМА:**





