Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Основы алгоритмизации и программирования (ОАиП)

ОТЧЕТ

по разминке №5

Тема работы: Длинная арифметика: вычитание

Выполнил

студент: гр. 251003 Панкратьев Е.С.

Проверил: Фадеева Е.П.

Минск 2022

**СОДЕРЖАНИЕ**

[1 Исходная постановка задачи 3](#_Toc120385779)

[2 Дополненная постановка задачи 4](#_Toc120385780)

[3 Методика решения 5](#_Toc120385781)

[3.1 Описание оператора try..except..end и функции Pos 5](#_Toc120385782)

[3.1.1 Оператор try..except..end 5](#_Toc120385783)

[3.1.2 Функция Pos и использование константной строки 5](#_Toc120385784)

[3.2 Условия ввода 5](#_Toc120385785)

[3.3 Проверка введенных данных и запись числа в массив 6](#_Toc120385786)

[3.3.1 Проверка ввода системы счисления и количества чисел для вычитания 6](#_Toc120385787)

[3.3.2 Проверка корректности введенного числа и его запись в массив 6](#_Toc120385788)

[3.4 Вычитание чисел 6](#_Toc120385789)

[3.5 Вывод разности всех чисел 7](#_Toc120385790)

[4 Структура данных 8](#_Toc120385791)

[Приложение А 9](#_Toc120385792)

[Приложение Б 19](#_Toc120385793)

# Исходная постановка задачи

Даны два целых числа в десятичной системе счисления до 50 цифр, которые >=0. Необходимо найти их разность.

# Дополненная постановка задачи

Вводится необходимая система счисления NS (максимальная система счисления – 20-ая) и количество чисел для вычитания n (не больше 50). Числа должны быть целыми, не более 256 символов и >=0. Необходимо найти их разность.

# [Методика решения](#_Toc83996305)

## Описание оператора try..except..end и функции Pos

### Оператор try..except..end

Оператор try..except..end имеет вид:

try  
 операторы;  
except  
  блок обработки исключений;  
end;

Выполнение блока начинается с секции try, при отсутствии исключительных ситуаций только она и выполняется. Секция except получает управление в случае возникновения исключительной ситуации. После обработки выполняются операторы, стоящие после end.

### Функция Pos и использование константной строки

Функция Pos имеет вид:

Pos(Substr, S)

Функция возвращает индекс символа, с которого начинается первое вхождение подстроки Substr в строку S. Eсли заданная подстрока не встречается в заданной строке, то функция вернет 0.

В данной задачи в роли подстроки Substr применяется символ из введенной строки; в роли строки S применяется константная строка, в которой индекс элемента уменьшенный на единицу соответствует численному значению символа в этом элементе:

NSAlphabet = ''

## Условия ввода

Условия ввода:

* система счисления NS должна быть целочисленной, больше либо равно 2 и меньше либо равно 20;
* количество чисел для вычитания n должно быть целочисленным, больше 1 и меньше либо равно 50;
* при вводе чисел, длина должна быть не больше 256 символов и значение каждого символа должно соответствовать введенной системе счисления NS.

## Проверка введенных данных и запись числа в массив

### Проверка ввода системы счисления и количества чисел для вычитания

Проверка ввода системы счисления NS и количества чисел для вычитания n происходит с помощью цикла с предусловием repeat..until, чтобы при вводе некорректных данных пользователь заново заполнял их. В теле цикла с помощью оператора try..except..end (описание оператора см. [главу 3.1.1](#_Оператор_try..except..end)) проверяем целочисленность; оператором if проверяет принадлежность заданному промежутку.

### Проверка корректности введенного числа и его запись в массив

Проверка корректности введенного числа происходит с помощью цикла с предусловием repeat..until, чтобы при вводе некорректных данных пользователь заново заполнял их. В теле цикла с помощью оператора if проверяем условие: длина должна быть не больше 256 символов.

Так в данной задаче необходимо работать с очень большими числами (которые не входят ни в один тип данных), необходимо записывать каждую цифру числа в отдельный элемент массива. Для удобства будем записывать числа с конца.

Перед началом записи числа в массив заполним все его элементы 0 с помощью цикла с параметром for. Далее для записи числа в массив используем цикл с предусловием while, в котором условие входа – i <= len и flag = false (i – счетчик цикла, len – длина введенной строки, flag – индикатор проверки правильности ввода). В теле цикла для перевода из символа в его численное значение используем константную строку NSAlphabet (в которой индекс элемента уменьшенный на единицу соответствует численному значению символа в этом элементе) и процедуру Pos (описание процедуры и константной строки см. [главу 3.1.2](#_Функция_Pos_и)). Далее проверяем корректность (символ должен соответствовать введенной системе счисления) с помощью оператора if. Если соответствует, то численное значение символа записываем в элемент массива.

## Вычитание чисел

Вычитание чисел происходит поэтапно: сначала находится разность двух чисел, потом разность между разностью предыдущих двух чисел и третьим числом, потом между разностью предыдущих чисел и следующим числом и так далее.

Вычитание двух чисел происходит с помощью метода вычитания столбиком. Примечание: перед нахождением разности двух чисел надо сравнить их и от большего числа вычесть меньшее, и если большим числом окажется вычитаемое, то в результате появится отрицательное число, и в дальнейшем, если еще надо проверисти операции вычитания, надо складывать числа (-a-b=

-(a+b)), подробнее об сложении чисел см. [ReportForWarmUp4](../WarmUp4/ReportForWarmUp4.docx).

Так как числа записаны с конца, вычитать элементы начнем сначала. Если в результате вычитания элементов получилось отрицательное число, то мы забираем единицу со следующего элемента и текущему добавляем выбранную систему счисления NS.

## Вывод разности всех чисел

Для перевода из численного значения символа в символ используем константную строку NSAlphabet, так как данная строка хранит символы, в которых индекс элемента уменьшенный на единицу соответствуют численному значению символа в этом элементе.

Разность записана с конца, значит вывод элементов массива следует начинать с конца. Для вывода разности используем цикл с параметром for. Выводить символы будем из константной строки NSAlphabet по индексу текущего значения элемента из массива разности, увеличенного на единицу. Примечание: если после разности чисел появились ненужые нули, то их выводить не нужно и если после вычитаний появился знак минус, то выводим его перед началом записи разности чисел.

# Структура данных

Таблица 1 – Данные

|  |  |  |
| --- | --- | --- |
| Элементы данных | Рекомендуемый тип | Назначение |
| MaxSize | Integer | Максимальный допустимый размер числа |
| MaxAmount | Integer | Максимальное допустимое количество чисел для вычитания |
| NSAlphabet | String | Набор символов для перевода символа в его численное значение и наоборот. |
| Num1 | Array [1..(MaxSize + (MaxAmount div 10) + 1)] of ShortInt | Массив первого числа |
| Num2 | Array[1..MaxSize] of ShortInt | Массив второго числа |
| str | String | Строка для записи числа в массив |
| MaxNS | ShortInt | Максимальная допустимая система счисления |
| NS | Integer | Введенная система счисления |
| Amount | Integer | Введенное количество чисел для вычитания |
| j | ShortInt | Счетчик цикла |
| Res | ShortInt | Разность цифр текущего разряда |
| Carry | ShortInt | Перенос в следующий разряд |
| Sign1 | ShortInt | Знак (для вычитания) первого числа |
| Sign2 | ShortInt | Знак (для вычитания) второго числа |
| Len1 | Word | Длина первого числа |
| Len2 | Word | Длина второго числа |
| Size | Word | Длина разности двух чисел |
| i | Word | Счетчик цикла |
| flag | Boolean | Индикатор проверки правильности ввода |
| Minus | Boolean | Индикатор появления минуса в результате разности |
| FoundLarger | Boolean | Индикатор нахождения большего числа |
| ZerosRemoved | Boolean | Индикатор удаления ненужный нулей |

Приложение А

(обязательное)

Исходный код программы

Program WarmUp5;

{

The program should calculate the residual of n numbers

in the entered number system

}

{$APPTYPE CONSOLE}

Uses

System.SysUtils;

Const

MaxSize=256;

MaxAmount=50;

NSAlphabet = '0123456789ABCDEFGHIJ';

MaxNS = length(NSAlphabet);

//MaxSize - maximum amount of digits for entered

//numbers

//MaxAmount - maximum amount of numbers

//NSAlphabet - transfer between symbols and numbers

//MaxNS - maximum number system

Var

Num1 :array[1..(MaxSize + (MaxAmount div 10) + 1)] of

ShortInt;

Num2 :array[1..MaxSize] of ShortInt;

str :string;

NS, Amount : Integer;

j, Res, Carry, Sign1, Sign2 : ShortInt;

Len1, Len2, Size, i : Word;

Minus, ZerosRemoved, FoundLarger, flag: boolean;

//Num1 - array of digits of the first number

//(further the residual of the two previous numbers)

//Num2 - array of digits of the second numbers (with

//which need to subtract the first number)

//str - string variable for writing numbers

//NS - number system

//Amount - amount of numbers

//Res - residual of digits

//Carry - сarry 1 (if there is) to the next element

//Sign1, Sign2 - the variable stores the sign in order

//to subtract the fewer number from the larger number

//Len1 - the length of the first entered number

//Len2 - the length of the second entered number

//Size - the size of the residual

//i, j - cycle counter

//Minus - if b is greater than a, then the answer will

//be with a minus

//ZerosRemoved - variable to remove the unnecessary

//zeros (if there is)

//FoundLarger - when a larger number is found, the

//variable will become true

//flag - flag to confirm the correctness of entering

//numbers

Begin

Writeln('Enter number system (maximum number system is

',MaxNS,', minimal is 2). The program will

calculate their residual.');

//Cycle with postcondition for entering correct data.

Repeat

//Initialize the flag

flag:= False;

//Validating the correct input data type

Try

Readln(NS);

Except

Writeln('Wrong input of number system! It must be

an integer');

flag:= True;

End;

//Validate Range

if ((NS > MaxNS) or (NS < 2)) and (flag = False) then

begin

Writeln('Wrong input of number system! It must be

>=2 and <=',MaxNS);

flag:= True;

end;

Until flag = False;

Writeln;

Writeln('Enter amount of numbers (no more than ',

MaxAmount,' and more than 1)');

//Cycle with postcondition for entering correct data.

Repeat

//Initialize the flag

flag:= False;

//Validating the correct input data type

Try

Readln(Amount);

Except

Writeln('Wrong input of amount of numbers! It must

be an integer');

flag:= True;

End;

//Validate Range

if ((Amount > MaxAmount) or (Amount <= 1)) and (flag

= False) then

begin

Writeln('Wrong input of amount of numbers! It must

be >1 and <=',MaxAmount);

flag:= True;

end;

Until flag = False;

//Declaring available symbols and their value

Writeln;

Writeln('Available symbols on the ',NS,'th number

system and their number system:');

for i := 0 to (NS - 1) do

Writeln('Symbol ', NSAlphabet[i+1],' Value = ',i);

Writeln;

Writeln('Enter numbers (no more than ',MaxSize,'

digits).');

//Cycle with postcondition for entering correct data.

Repeat

//Initialize the flag

flag:= False;

//Read the first entered number and check for

//correctness.

Readln(str);

//Find length of the first number

Len1:= length(str);

//Checking the correct length

if Len1 > MaxSize then

begin

Writeln('Wrong input of number! The length of the

number must be no more than ',MaxSize,'

digits.');

flag:= True;

end

//Else if length >1, the first digit cannot be 0

//(in the mirrored view it is last)

else if (Len1 > 1) and (str[1] = '0') then

begin

Writeln('Wrong input of number! The first digit of

a number cannot be 0');

flag:= True;

end

//Else writing a number to an array and checking for

//valid symbols

else

begin

//Reset the first number for the input

for i := 1 to length(Num1) do

Num1[i]:= 0;

//Write the first entered number in mirrored view

//to an array

i:=1;

while (i <= Len1) and (flag = False) do

begin

//Transfer to numerical value (-1 because

//numbering in delphi starts from 1)

Num1[i]:= Pos(str[Len1-i+1], NSAlphabet) - 1;

//Checking for correct input in the NS. Num1[i]

//will be <0 if the symbol is not in NSAlphabet

if (Num1[i] < 0) or (Num1[i] >= NS) then

begin

Writeln('Wrong input of number! Namely, wrong

input of symbols! See available symbols

above!');

flag:= True;

end;

//Modernize i

i:= i + 1;

end;

end;

Until flag = False;

//The cycle go (Amount-1) times to subtract all the

//numbers

j:=1;

for j := 1 to (Amount - 1) do

begin

Writeln('-');

//Cycle with postcondition for entering correct

//data.

Repeat

//Initialize the flag

flag:= False;

//Read the second entered number and check for

//correctness.

Readln(str);

//Find length of the second number

Len2:= length(str);

//Checking the correct length

if Len2 > MaxSize then

begin

Writeln('Wrong input of number! The length of

the number must be no more than

',MaxSize,' digits.');

flag:= True;

end

//Else if length >1, the first digit cannot be 0

//(in the mirrored view it is last)

else if (Len2 > 1) and (str[1] = '0') then

begin

Writeln('Wrong input of number! The first digit

of a number cannot be 0');

flag:= True;

end

//Else writing a number to an array and checking

//for valid symbols

else

begin

//Reset the second number for the input

for i := 1 to length(Num2) do

Num2[i]:= 0;

//Write the second number in mirrored view to an

//array

i:=1;

while (i <= Len2) and (flag = False) do

begin

//Transfer to numerical value (-1 because

//numbering in delphi starts from 1)

Num2[i]:= Pos(str[Len2-i+1], NSAlphabet) - 1;

//Checking for correct input in the NS. Num2[i]

//will be <0 if the symbol is not in NSAlphabet

if (Num2[i] < 0) or (Num2[i] >= NS) then

begin

Writeln('Wrong input of number! Namely,

wrong input of symbols! See available

symbols above!');

flag:= True;

end;

//Modernize i

i:= i + 1;

end;

end;

Until flag = False;

//The larger size assign in a Size (this is the size

//of the residual of the numbers)

if Len1 > Len2 then

Size:= Len1

else

Size:= Len2;

//Searching the larger number. To do this, compare

//the first digits (in the mirrored view it is

//last). If they are equal, move on. If the residual

//is negative, then there is no need to compare

//further (since will be the sum of negative numbers)

i:= Size;

FoundLarger:= False;

while (i >= 1) and (FoundLarger = False) and (Minus

= False) do

begin

//If Num1>Num2, the sign for the first number is

//'+', for the second '-' (Num1-Num2).

if Num1[i] > Num2[i] then

begin

Sign1:= 1;

Sign2:= -1;

//Exit the cycle because found a larger number

FoundLarger:= True;

end

//If Num2>Num1, the sign for the first number is

//'-', for the second '+' (Num2-Num1).

else if Num2[i] > Num1[i] then

begin

Sign2:= 1;

Sign1:= -1;

//Variable Minus since the answer is negative

//(Num1<Num2)

Minus:= True;

//Exit the cycle because found a larger number.

FoundLarger:= True;

end;

//Modernize i

i:= i - 1;

end;

//This condition is needed if the user entered two

//identical numbers. Their subtraction will be 0.

if (FoundLarger = False) and (Minus = False) then

begin

//Since the numbers are the same, the answer will

//be 0

for i := 1 to Len1 do

Num1[i]:= 0;

//The size of the subtract will be 1 (0 is one

//digit).

Size:=1;

//Note that compared numbers and supposedly found

//the largest

FoundLarger:= True;

end;

//In the cycle will subtract the numbers separately

for i := 1 to Size do

begin

//If found the largest number, then need to do the

//subtract of numbers

if (FoundLarger = True) then

begin

//Considering the signs, starting to subtract

//the last digits of the numbers (in the mirrored

//view it is first)

//and consider the carry (if there is).

Res:= Sign1\*Num1[i] + Sign2\*Num2[i] + Carry;

//If the difference is less than zero, then take

//1 (for the current digit it is NS) from the

//next digit

if Res < 0 then

begin

Res:= Res + NS;

Carry:= -1;

end

//Else carry = 0

else

Carry:= 0;

//Put the result in the residual array c

Num1[i]:= Res;

end

//If did not compare the numbers, then the first

//and second numbers are negative, need to add them

else

begin

//Starting to add (add negative numbers) the last

//digits of the numbers (in the mirrored view it

//is first)

//and add the carry (if there is).

Res:= Carry + Num1[i] + Num2[i];

//The modulo of the Sum by number system is the

//last(in the mirrored it is first) digit of sums

//(array Num1).

Num1[i]:= Res mod NS;

//The integer part of dividing by number system

//is the carry that will go to the next element

Carry:= Res div NS;

//If there is a carry on the last digit, then

//increase the size of sum by 1 and add a carry

//to the next element

if (i = Size) and (Carry = 1) then

begin

Size:= Size + 1;

Num1[i+1]:= Carry;

end;

end;

end;

//Update the length of the first number

Len1:= Size;

//Reset the carry for the following operations

Carry:= 0;

end;

Writeln('The residual of the numbers is:');

//If the second number is greater than the first, then

//write the sign -

if Minus = True then

Write('-');

//If size = 1, no need to remove zeros

if size = 1 then

ZerosRemoved:= True;

//Write the answer, mirroring the back. Аlso remove

//unnecessary zeros(if there is)

//And transfer to symbolic value

for i := Size downto 1 do

begin

if (ZerosRemoved = False) and (Num1[i] > 0) then

ZerosRemoved:= True;

if (ZerosRemoved = True) then

Write(NSAlphabet[Num1[i]+1]);

end;

Readln;

End.

Приложение Б

(обязательное)

Тестовые наборы

**Тестовая ситуация: некорректный ввод данных**

Тест 1

Исходные данные: Некорректный ввод системый счисления

Ожидаемый результат: Повторная попытка

Полученный результат:

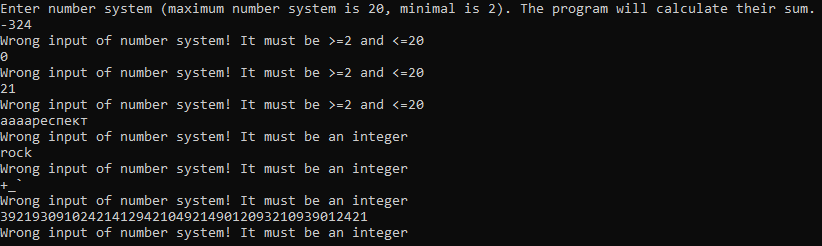


Рисунок – Результаты расчетов

Тест 2

Исходные данные: Некорректный ввод количества чисел для вычитания

Ожидаемый результат: Повторная попытка

Полученный результат:

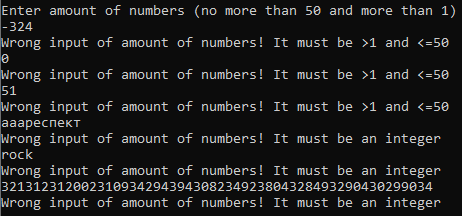


Рисунок – Результаты расчетов

Тест 3

Исходные данные: Некорректный ввод первого числа

Ожидаемый результат: Повторная попытка

Полученный результат:

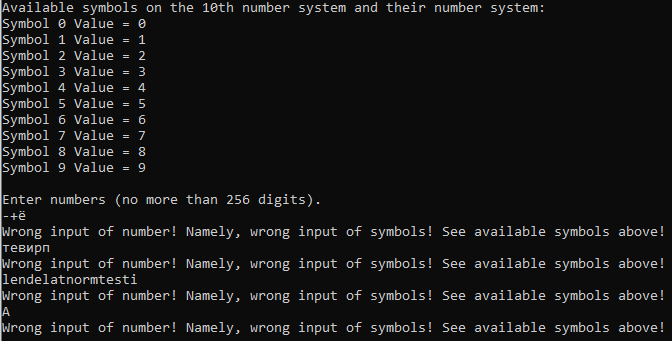


Рисунок – Результаты расчетов

Тест 4

Исходные данные: Некорректный ввод второго числа

Ожидаемый результат: Повторная попытка

Полученный результат:

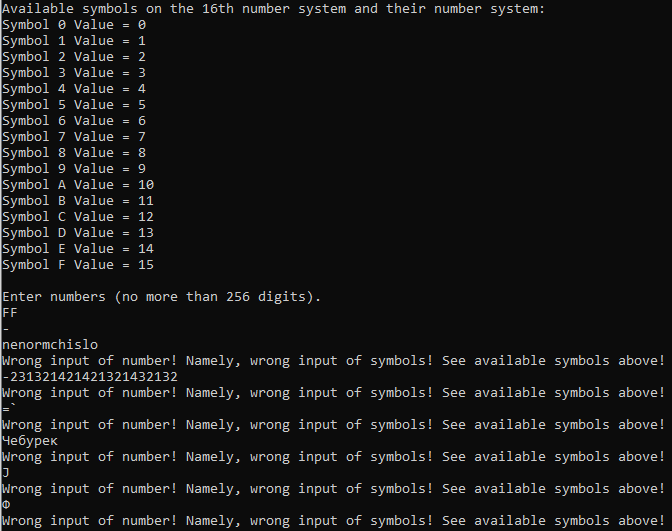


Рисунок – Результаты расчетов

**Тестовая ситуация: корректный ввод данных**

Тест 5

Исходные данные: Система счисления: 16; количество чисел для вычитания: 2.

Ожидаемый результат:

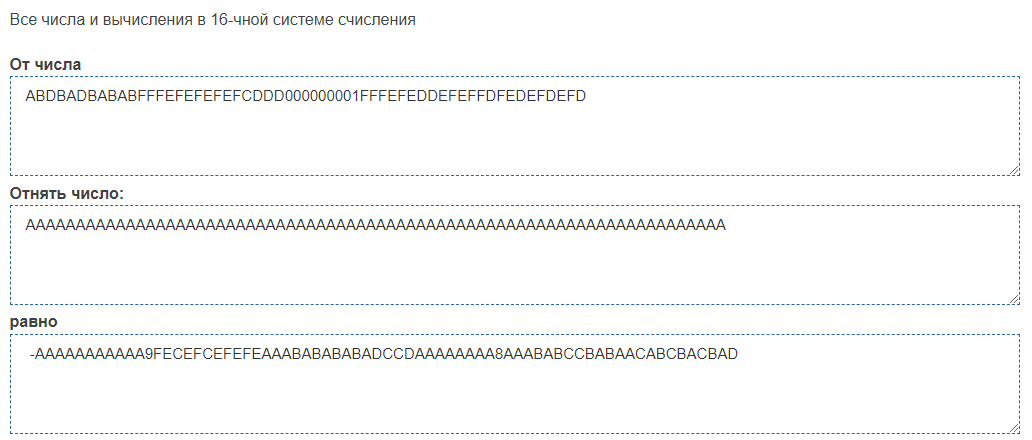


Рисунок – Ожидаемый результат (ilovecalc)

Полученный результат:

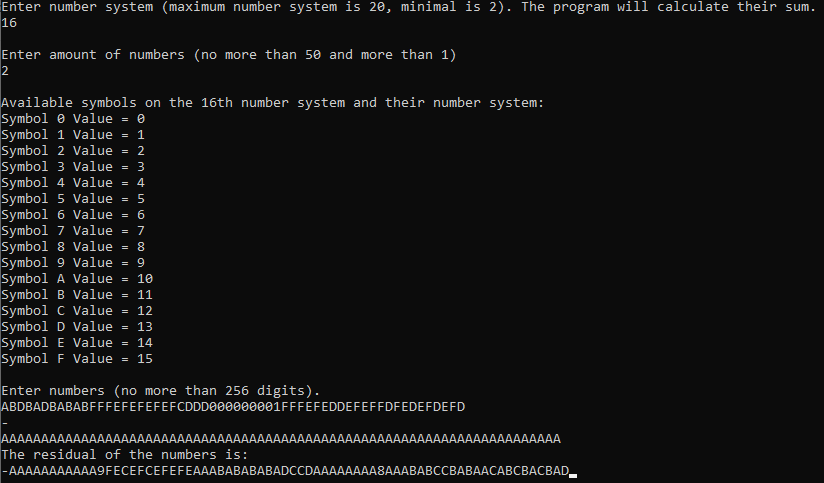


Рисунок – Результаты расчетов

Тест 6

Исходные данные: Система счисления: 2; количество чисел для вычитания: 2.

Ожидаемый результат:

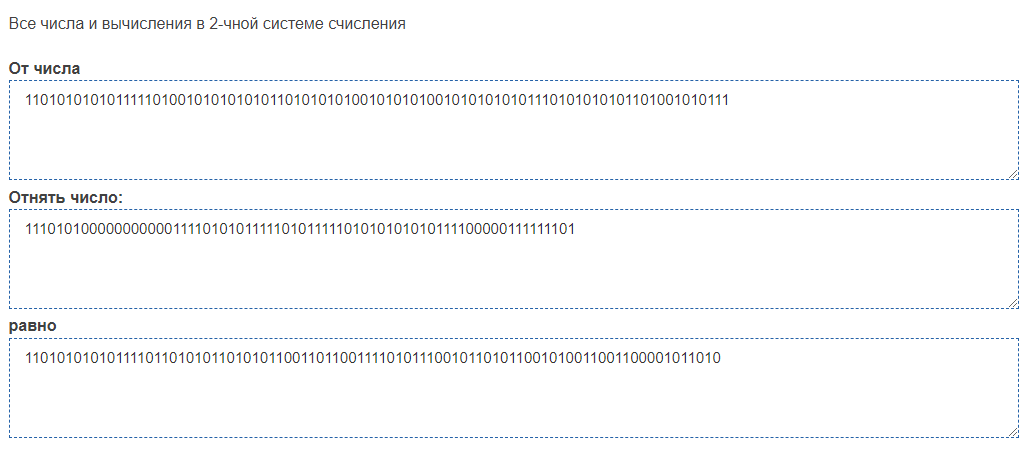


Рисунок – Ожидаемый результат (ilovecalc)

Полученный результат:

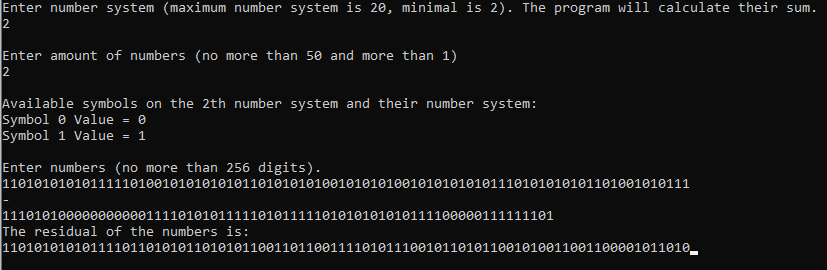


Рисунок – Результаты расчетов

Тест 7

Исходные данные: Система счисления: 20; количество чисел для вычитания: 2.

Ожидаемый результат:

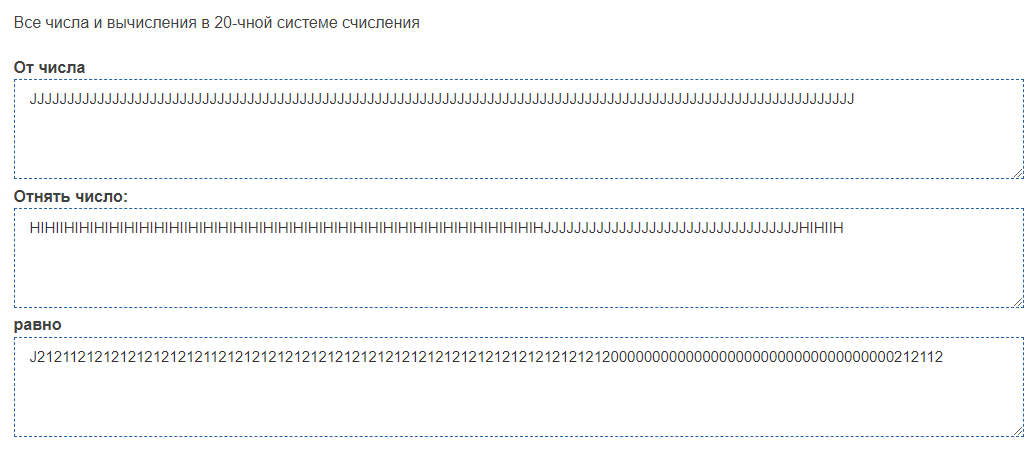


Рисунок – Ожидаемый результат (ilovecalc)

Полученный результат:

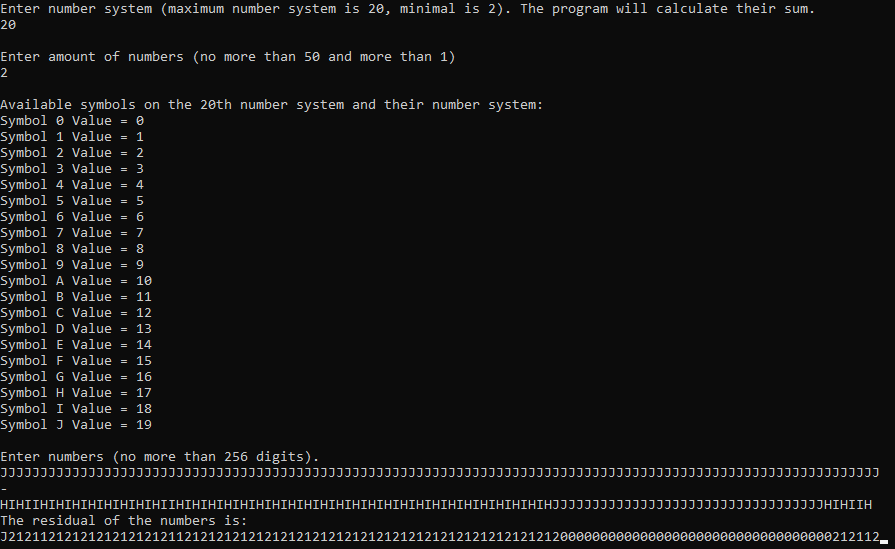


Рисунок – Результаты расчетов

Тест 8

Исходные данные: Система счисления: 10; количество чисел для вычитания: 2.

Ожидаемый результат:

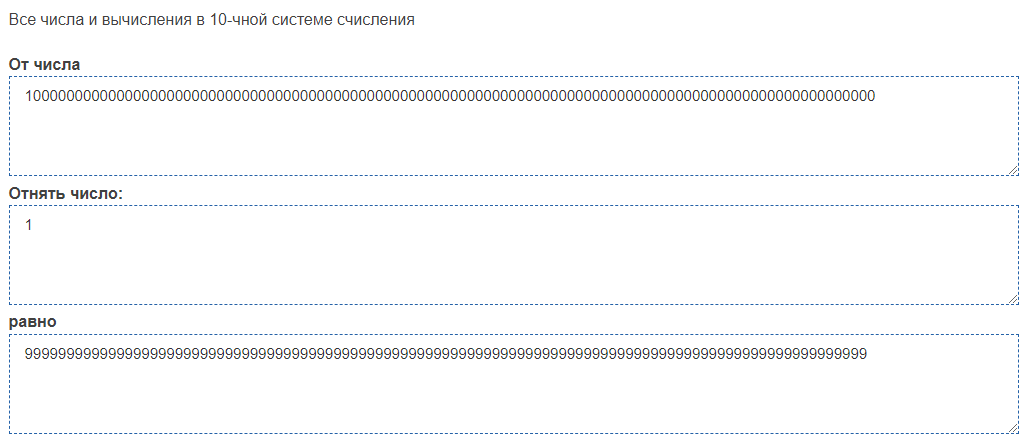


Рисунок – Ожидаемый результат (ilovecalc)

Полученный результат:

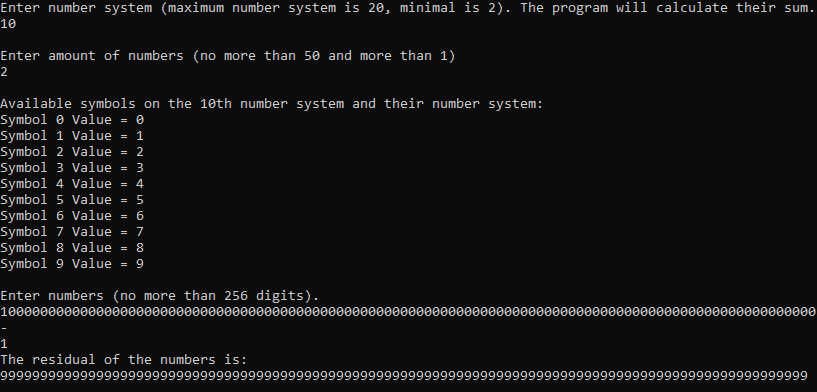


Рисунок – Результаты расчетов

Тест 9

Исходные данные: Система счисления: 10; количество чисел для вычитания: 6.

Ожидаемый результат:

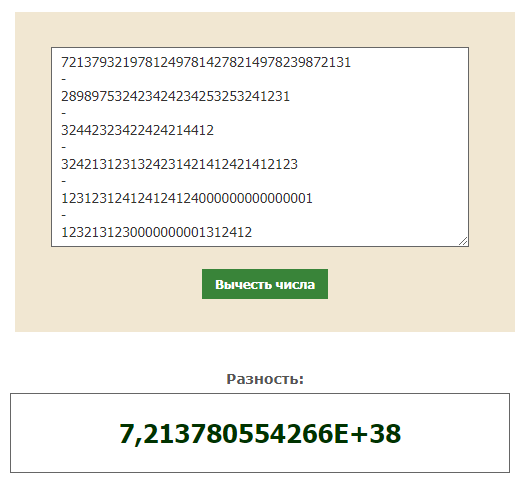


Рисунок – Ожидаемый результат (tools.mega8)

Полученный результат:

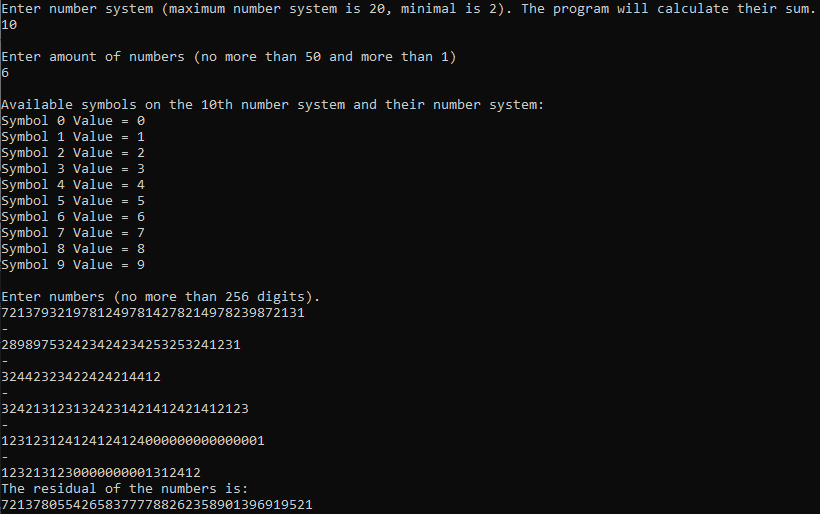


Рисунок – Результаты расчетов