**Отчет курсанта Громова Григорий Андреевича группы 22.Б05 о выполнении практического задания на 03.11.2022**

***Задания на рекурсию***



**Алгоритм (более подробный алгоритм смотрите в комментариях к тексту программы):**

1. В main создаем переменные n и a
2. **n** (число, сумму цифр которого нужно найти) инициализирует пользователь через клавиатуру, **a** присваиваем значение рекурсивной функции rec1, передавая в качестве параметра **n**
3. Выводим **а** на экран
4. Рекурсивная функция, получив число **n1**, возвращает *само число*, если оно однозначное и *сумму последней цифры числа и значения самой себя от числа без последней цифры*, если число не однозначное

**Математическая модель:** очевидна

**Текст программы:**

#include <stdio.h>

#include <stdlib.h>

int rec1(int n1); //объявляем рекурсивную функцию

int main(){

int n, a; //заводим локальные переменные

printf("Введите натуральное число: ");

scanf("%d", &n); //инициализируем n

a = rec1(n); //инициализируем a (вызываем функцию rec1, передавая ей значение переменной n)

printf("Сумма его цифр равна %d\n", a); //выводим результат на экран

}

//рекурсивная функция

int rec1(int n1){

if (n1 > 9){ //если число двухзначное или длиннее

return n1 % 10 + rec1(n1 / 10); //возвращаем сумму его последний цифры и значения функции rec1

//от этого числа, деленного на 10

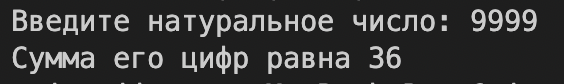
} else{

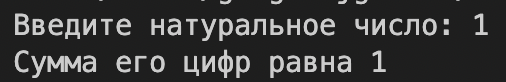
return n1; //в противном случае (то есть если число однозначное) возвращаем само число

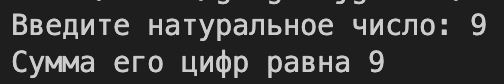
}

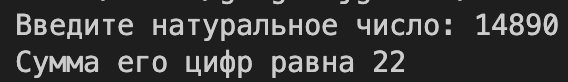
}

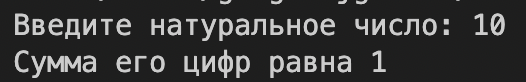
Скриншоты примеров работы:

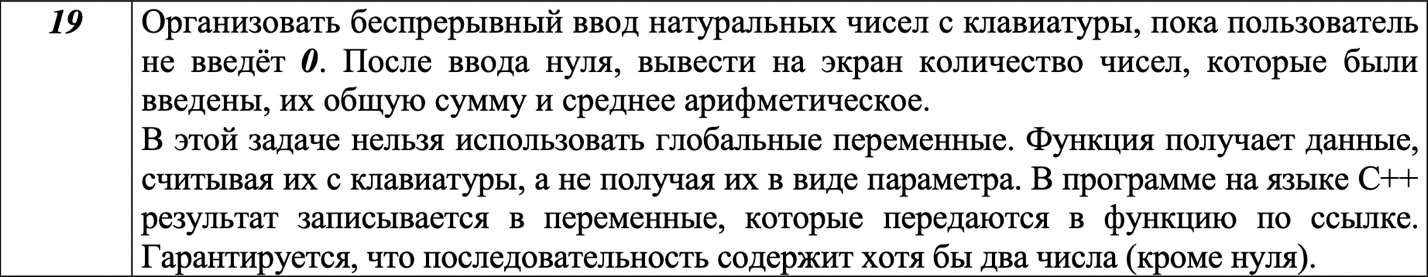












**Замечание:** соблюдая все условия задачи (то есть отсутствие глобальных переменных, параметров в функции и т.д.), у меня получилось только найти сумму введенных элементов. Ведь при соблюдении всех этих требований рекурсивная функция должна возвращать 2 числа – сумму и количество, а как это реализовать я не понимаю

**Алгоритм:**

1. В main создаем переменную sum и присваиваем туда значение, полученное из функции rec2(), выписываем ее
2. В rec2() мы создаем локальную переменную sum = 0
3. Cканируем введенное пользователем число и, если оно равно 0, возвращаем 0. В противном случае прибавляем к sum сумму *этого числа* и того, что *вернет функция rec2()*
4. Возвращаем sum

**Математическая модель:** очевидна

**Текст программы:**

#include <stdio.h>

#include <stdlib.h>

int rec2(); //объявляем рекурсивную функцию

int main(){

int sum;

sum = rec2();

printf("%d", sum);

}

int rec2(){

int n;

int sum = 0;

scanf("%d", &n);

if (n == 0){

return 0;

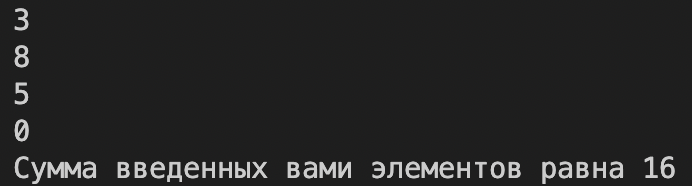
}

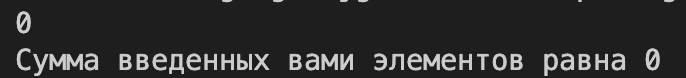
sum += (n + rec2());

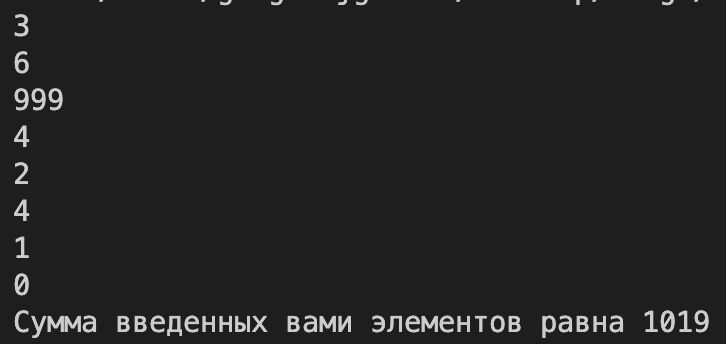
return sum;

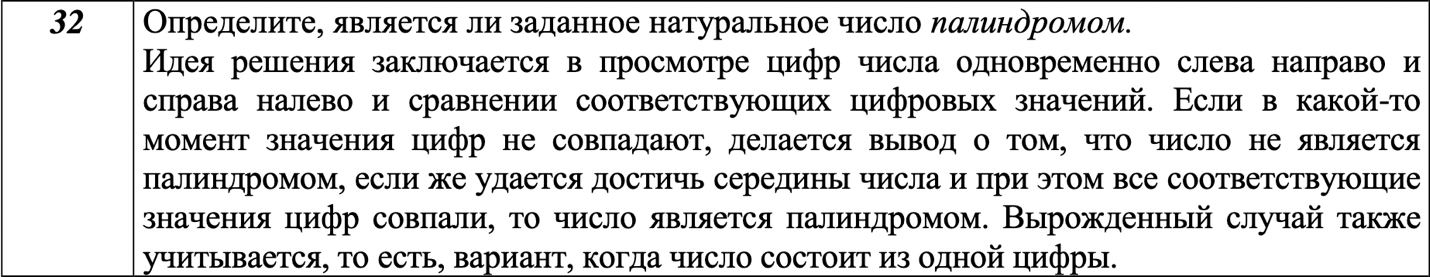
}

Скриншоты примеров работы:









**Алгоритм (более подробный алгоритм смотрите в комментариях к тексту программы):**

1. Интересующее нас посимвольно раскладываем в последовательные ячейки массива
2. Передаем этот массив на проверку на палиндром функции rec3(), она
3. Сравнивает элементы крайних ячеек, «срезает» их и заново сравнивает, «срезает»… до тех пор, пока они либо не будут различны, либо номер ячейки левого конца не превысит номер ячейки правого
4. В первом случае вернем 0 (вы ввели не палиндром), во втором 1 (это палиндром)

Текст программы:

#include <stdio.h>

#include <stdlib.h>

int len(int n); //функция, возвращающая количество разрядов натурального числа

int rec3(int M[], int start, int end); //функция, сравнивающая первый и последний элемент массива,

//возвращает 1, если они равны и 0, если отличаются

int main(){

int n;

printf("Введите натуральное число: ");

scanf("%d", &n);

int size = len(n);

int M[size]; //создаем массив с количеством ячеек, равным количеству разрядов в числе n

int i = 0; //вспомогатльная переменная для работы в цикле

//запишем в каждую ячейку массива по одной цифре числа

//при такой реализации порядок будет обратным, но для поиска палиндрома это не является проблемой

int last;

while (n > 0){

last = n % 10;

n = n / 10;

M[i] = last;

i++;

}

int a;

a = rec3(M, 0, size - 1); //ответ на воапрос является ли n палиндромом мы получаем от функции rec3()

if (a == 1){

printf("Введенное число является палиндромом\n");

}

else{

printf("Введенное число не является палиндромом\n");

}

}

int rec3(int M[], int start, int end){ //принимает копию массива, номер левой ячейки, номер правой ячейки

int a;

if (start < end){ //пока не дошли до середины массива

if (M[start] == M[end]){ //если содержимое ячеек совпадает

a = rec3(M, start + 1, end - 1); //смотрим является ли палиндромом "число" (а вернее строка, так как,

//например, 010 числом не является) без первой и последей цифр

return a; //возвращаем да или нет

}

else{ //если не совпали, возвращаем 0

return 0;

}

}

return 1; //если мы дошли до конца ни разу не получив 0, значит перед нам палиндром

}

int len(int n){ //очевидно

int counter = 0;

while (n > 0){

counter++;

n /=10;

}

return counter;

}

Скриншоты примеров работы:

