МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСІТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТІТУТ»

КАФЕДРА МЕДИЧНОЇ КІБЕРНЕТИКИ ТА ТЕЛЕМЕДИЦИНИ

**Лабораторна робота № 9**

з дисципліни «Алгоритмізація та програмування»

на тему: “ Рекурсія ”

Варіант 16

**Виконав:**

Студент групи БС-32

Хоміцевич Микита

**Перевірила:**

викл.

Матвійчук А.О.

Київ-2014

**Задание**

****

**□ Лабораторна робота виконана без зауважень**

**□ Лабораторна робота має зауваження:**

**□ присутні зауваження до блок-схем:**

**□ виконані не за стандартом**

**□ не відповідають коду**

**□ присутній код**

**□ інші зауваження:**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**□присутні зауваження до коду:**

**□задача завдання вирішена хибно**

**□ код програми не компілюється**

**□ використано глобальні змінні**

**□ недостатня декомпозиція на функціонуванні користувача**

**□ статичні змінні при роботі з масивами**

**□ оформлення коду**

**□присутні зайві символи «{»та «}»**

**□інші зауваження:**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**□невірні відповіді на запитання:**

**□№1 □№2 □№3**

**□маються інші зауваження: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Программный код**

#include <iostream>

#include <cstdio>

#include <cmath>

#include <conio.h>

#include <locale.h>

void MAIN();

double recurs(double n);

int main() {

MAIN();

\_getch();

}

void MAIN() {

setlocale(LC\_ALL, "RUS");

double n, sum;

std::cout << "Введите пожалуйста парное число сомножителей N: ";

std::cin >> n;

while ((int)n < 2 || ((int)n % 2 != 0)) {

std::cout << endl << "Введите ПОЖАЛУЙСТА парное число сомножителей N: ";

std::cin >> n;

}

sum = recurs(n);

std::cout << “Наше значение последовательности: ” << sum << std::endl;

}

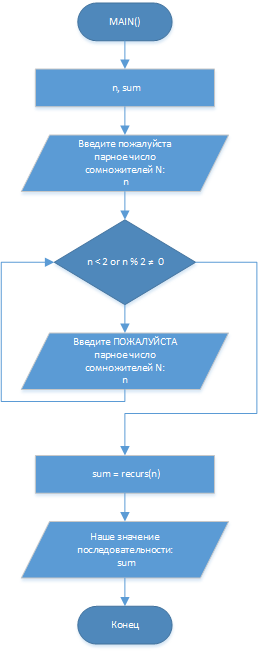
double recurs(double n) {

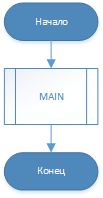
if (n > 2) return (recurs(n - 2) \* (n / (n - 1)) \* (n / (n + 1)));

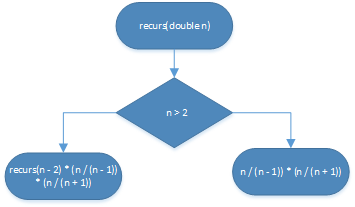
else return (n / (n - 1)) \* (n / (n + 1));

}

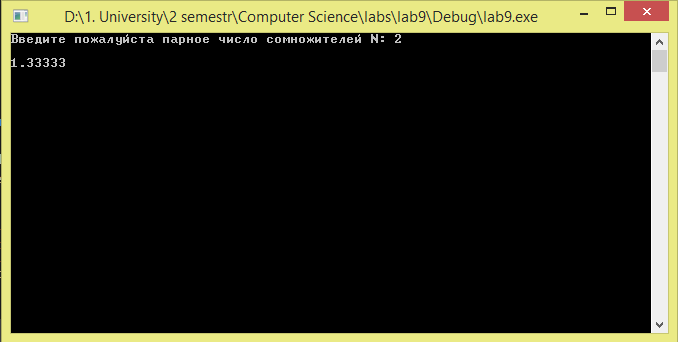
**Блок-схема**

****

****

****

**Скриншот работы программы**

****

**Ответы на контрольные вопросы**

1. Дайте определение рекурсии и приведите примеры ее использования в разных отраслях знаний?

Рекурсия - вызов [функции](http://ru.wikipedia.org/wiki/%D0%A4%D1%83%D0%BD%D0%BA%D1%86%D0%B8%D1%8F_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) ([процедуры](http://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%B4%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0#.D0.92.D0.B8.D0.B4.D1.8B_.D0.BF.D0.BE.D0.B4.D0.BF.D1.80.D0.BE.D0.B3.D1.80.D0.B0.D0.BC.D0.BC)) из неё же самой, непосредственно (*простая рекурсия*) или через другие функции (*сложная* или *косвенная рекурсия*), например, функция A вызывает функцию B, а функцияB — функцию A.

К примеру, в физике рекурсией может обозначится взаимное отображение двух стекл, что стоят друг на против друга и каждое отражает другое само в себе.

1. В каких задачах разумно использовать рекурсивные функции?

Это зависит от специфики самой задачи. Как правило, любой код можно оформить в виде прохода циклом (итеративно), что более приемлимо для понимания и меньше затратно по памяти. Но если необходимо где-то специально сократить код или рекурсивный вариант легче, чем итеративный, то лучше использовать рекурсию.

1. Что общего и в чем разница между циклическими и рекурсивными способами организации расчетов?

Схожесть:

1. Количество шагов (вложеность) для одной задачи одинакова
2. При наличии хвостовой рекурсии (констаного значения выделяемой памяти компилятором) кол-во итераций в цикле и кол-во вызовов рекурсии будет одинаково

Различия:

1. Рекурсивный метод часто более компактен
2. Теративный способ менее требователен к памяти
3. Легкость проверки коректности значений в рекурсии
4. Однозначность назначения каждой переменной в рекурсии
5. Рекурия выполняется определенное вложенное количество раз (как правило, не больше 50 вызовов)
6. Определить основные особенности работы рекурсивных функций в языке программирования С++

В рекурсии обязательно должно быть возвращаемое значение, как и в любой not-void фукнции

Рекурсия имеет определенный предел вложенности (в зависимости от задачи, но как правило, это кол-во достигает ~50-ти)

Для выполнения рекурсивной функции происходит вызов стека памяти

1. Что такое прямая и непрямая (опосередкована) рекурсия?

Прямая рекурсия – функция вызывает сама себя определенное кол-во раз, пока не выполнится условие для ее завершения.

Косвенная рекурсия – функция А вызывает ф-ию В, которая вызывает дальше ф-ию А. И так далее, пока так же не выполнятся условия для прекращения рекурсий.

1. Когда выполняется завершение рекурсивных вызовов? Что такое рекурсивное зацикливание и для чего оно приводит?

Завершение рекурсивного вызова происходит когда ф-ия достигла такого условия, когда ф-ия возвращает не саму себя, а какое-то значение.

Рекурсивное зацикливание – когда не выолняется условие выхода из рекурсии и рекурсия начинает выполнятся бесконечное количество раз, что привод к огромным затвратам (утечке) памяти и в дальнейшем – аварийному завершению программы.

1. Что такое область действия переменных?

Это специальная область какого-то объекта, в котором будет находится какая-то переменная с определенным именем и значением.

Эта область нужна для того, что бы в разных местах программы использовать одинаковые имена переменных, но с разными значениями.

1. Как необходимо выполнить объявление переменной, что бы доступ к ней был возможен из любой ф-ии поточного файла программы?

Нужно объявить глобальную переменную, т.е. переменную, которая не будет находится ни в одной из функций программы.

1. Как изменяются значения локальных и глобальных переменных при использовании рекурсивных ф-ий?

Локальная переменная, которая используется в рекурсии, после выхода из нее (рекурсии) удаляется из памяти, в отличии от глобальной, значение которой изменяется и сохраняется дальше.

1. Определите главные ограничения при использовании рекурсий.
2. Необходимость возвращать какое-то значение
3. Глубина рекурсии
4. Вызов стека памяти
5. Снижение производительности