# НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із лабораторної роботи №*4*

із дисципліни «Програмування»

на тему

*«Вказівники, функція, рекурсія»*

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-63 | *асистент Громова В. В.* |
| *Артеменко Я.К.* |  |

Київ — 2017

# **ЗМІСТ**

[1 ПОСТАНОВКА ЗАДАЧІ 2](#_Toc478382182)

[1.1 Мета роботи 2](#_Toc478382183)

[1.2 Завдання на лабораторну роботу 2](#_Toc478382184)

[2 ОПИС ПРОГРАМИ 3](#_Toc478382185)

[ВИСНОВКИ 5](#_Toc478382186)

[Додаток А (блок-схема) 5](#_Toc478382187)

[Додаток Б (відповіді на контрольні запитання) 8](#_Toc478382188)

[Додаток В (текст програми) 9](#_Toc478382189)

# **1 ПОСТАНОВКА ЗАДАЧІ**

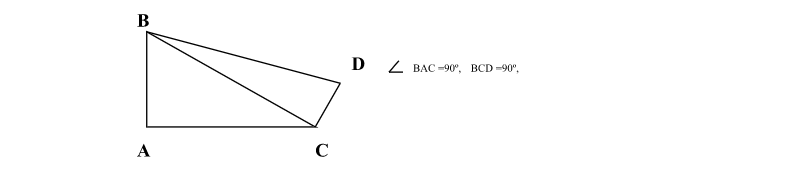
* 1. Мета роботи

Ознайомитися з новим типом організації даних С (вказівники) і набути практичних навичок його використання для написання програм; оволодіти синтаксисом написання функцій і їхніх прототипів, методикою складання і відлагодження програм, що містять функції, специфікою передачі параметрів у функцію та повернення одержаних результатів; вивчити методи використання алгоритмів і програм із рекурсією мовою С.

* 1. Завдання на лабораторну роботу

Варіант №19

1. завдання:
2. Знайти периметр фігури ABCD по заданим сторонам AB, AC і DC. Написати функцію для розрахунку гіпотенузи прямокутного трикутника по його катетам.



1. Дано натуральне число N. Вивести слово YES, якщо число N є точним ступенем двійки, або слово NO в протилежному випадку. Операцією зведення в степінь користуватись неможна!

Зробити 2 варіанта:

а) через рекурсію;

б) звичайною функцією.

* 1. Вимоги до лабораторної роботи

**Постановка першого завдання**

Програма повинна задовольняти такі вимоги:

1. Масиви можна генерувати автоматично, якщо інше не обумовлено в завданні. Решту вихідної інформації потрібно уводити з клавіатури на запит.
2. Працювати потрібно тільки через указівники та функції.
3. Усі значення функціям потрібно передавати як параметри.
4. На екрані обов’язково повинні зберігатися як вихідна інформація, так і результат.

**Постановка другого завдання:**

Виконуючи це завдання, потрібно скласти алгоритми та програми з використанням рекурсії відповідно до варіанту завдання.

При цьому потрібно враховувати граничні умови виходу з рекурсії.

# **ОПИС ПРОГРАМИ**

Дана програма містить 11 функцій: menu\_main, gip1, gip2, result\_gip, P, task\_1, no\_recursion, recursion, output\_recursion, task\_2, main.

Функція **menu\_main** виводить на екран дані про виконавця та надає можливість користувачу обрати завдання для виконання. Обрати завдання можна за допомогою стрілок «вверх» і «вниз» і за допомогою клавіші «Enter» перейти до його виконання. Пункти, які ми можемо обрати підсвічені білим кольором, а умови завдань виділяються сірим кольором. За допомогою оператора розгалуження «if – else» зроблено можливість переміщення «стрілки вибору» між пунктами меню та за допомогою десяткових індексів функціональних клавіш («80» – для стрілки вниз, а «72» – для стрілки вверх) зроблено можливим використання клавіш для вибору. За допомогою циклу «while» очікується натиснення клавіші Enter, після чого ця функція повертає значення вибраного пункту в функцію main.

Функція **gip1** знаходить гіпотенузу BC прямокутного трикутника АВС за допомогою значень катетів АВ і АС. Для цього використовується теорема Піфагора.

Функція **gip2** знаходить гіпотенузу BD прямокутного трикутника BCD за допомогою значень катетів BC і СD. Для цього використовується теорема Піфагора і береться значення сторони BC з функції **gip1**.

Функція **result\_gip** виводить на екран значення гіпотенуз BD і ВС. Для цього в самій функції ми викликаємо функції **gip1** та **gip2**.

Функція **Р** обчислює периметр фігури ABCD за сторонами AB, CD і АС, які вводяться з клавіатури, та сторони BD , яка обчислюється за допомогою функції **gip2**.

В функції **task\_1** за допомогою оператора **if** обирається одна із підзадач першої задачі і викликається відповідна функція.

Функція **no\_recursion** розв’язує другу задачу без рекурсії та у випадку, коли число задовольняє умові, то виводиться YES, а в іншому випадку NO.

Функція **recursion** представляє собою рекурсивну функцію, яка повертає значення TRUE у випадку, коли число задовольняє умові, та FALSE, якщо протирічить.

У функції **output\_recursion** викликається функція **recursion** та у випадку, коли число задовольняє умові, то виводиться YES, а в іншому випадку NO.

В функції **task\_2** за допомогою оператора **if** обирається одна із підзадач другої задачі і викликається відповідна функція.

Функція **main** є головною і в ній викликаються всі функції і забезпечується робота меню за допомогу оператора «case».

# **ВИСНОВКИ**

На цій лабораторній роботі здобуто навички по роботі з новим типом організації С (вказівники) і набуто практичні навички його використання для написання програм; здобуто навички по написанню функцій та їх прототипів, та методи використання програм з рекурсією в мові С.

# **Додаток А (блок-схема)**

Функція menu\_main:

Початок

System(“cls”);

Printf(“Інформація про виконавця”)

Printf(“Умова завдання”)

ні

If key = {0}

так

Cout<<”-> Продовжити”

Cout<<” Продовжити”

If key = {1}

ні

так

Cout<<”-> Вихід”

Cout<<” Вихід”

Кінець

Функція gip1 (функції gip2 і Р аналогічно):

Початок

Cout<<” Введіть сторони”

Scanf(сторони)

Кінець

Обчислення відповідної формули

Функція no\_recursion:

Кінець

cout << ((n & (n - 1)) ? "NO" : "YES") << endl;

Початок

Функція recursion:

If ((n == 2) || (recursion(n / 2)))

If (trunc(n) != ceil(n))

Початок

ні

так

Return False

так

Кінець

Return True

Функція output\_recursion:

If (recursion(n))

Початок

ні

так

Кінець

cout<<"NO"<<endl;

cout<<"YES"<<endl;

# **Додаток Б (відповіді на контрольні запитання)**

1. Вказівник – це адреса елемента пам’яті.
2. Для роботи з масивом дуже часто використовують вказівники.
3. Прототип функції можна вказувати замість її опису до моменту її виклику для того, щоб компілятор міг виконати перевірку відповідності типів аргументів і параметрів.
4. Параметри, які записують у зверненні до функції, називають фактичними, а параметри, зазначені в описі функції – формальними. Об’єкти, оголошені поза функцією, діють у будь-якій функції. Їх називають глобальними. Об’єкти, оголошені всередині функції, діють тільки в ній. Їх називають локальними.
5. За допомогою вказівника.
6. Вони нічого не повертають.
7. Рекурсія – це спосіб організації допоміжного алгоритму у вигляді підпрограми (процедури або функції), за якого вона в ході виконання звертається сама до себе.
8. Глобальні у всій програмі, а локальні лише в рекурсії.
9. Це умова, при якій рекурсія зупиниться.
10. Породження все нових копій рекурсивної функції до виходу на граничну умову називаються рекурсивним спуском.
11. Завершення роботи рекурсивних функцій, аж до найпершої, яка ініціювала рекурсивні виклики, називають рекурсивним підйомом.

# **Додаток В (текст програми)**

#include <stdio.h>

#include <locale.h>

#include <windows.h>

#include <conio.h>

#include <iostream>

#include <stdlib.h>

#include <iomanip>

#include <math.h>

using namespace std;

float bc\_2,bd\_2;

int menu\_main()

{

HANDLE hStdOut = GetStdHandle(STD\_OUTPUT\_HANDLE);

char key = 0;

int code;

do

{

system("cls");

printf("\t\t\t\t \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\t\t\t\t \* Лабораторная работа №4 \*\n");

printf("\t\t\t\t \* Указатели, \*\n");

printf("\t\t\t\t \* функции и рекурсии \*\n");

printf("\t\t\t\t \* Вариант №19 \*\n");

printf("\t\t\t\t \* Группа KM-63 Артеменко Я.К.\*\n");

printf("\t\t\t\t \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\n\n");

key = (key + 5) % 5;

SetConsoleTextAttribute(hStdOut, (WORD)((0 << 4) | 8));

printf("\tИНСТРУКЦИЯ!!! ДЛЯ НАВИГАЦИИ ПО МЕНЮ ИСПОЛЬЗУЙТЕ СТРЕЛКИ ВВЕРХ, ВНИЗ И КЛАВИШУ ENTER\n");

printf("\t1. Найти периметр фигуры АВСD по заданным сторонам AB,AC,DC. \n");

printf("\t Написать функцию для расчета гипотенузы прямоугольного треугольника \n");

printf("\t по его катетам. \n");

printf("\t\tB\*\n");

printf("\t\t \* \* \*\n");

printf("\t\t \* \* \*\n");

printf("\t\t \* \* \*\n");

printf("\t\t \* \* \*D угол прямой между BA и AC, BC и CD\n");

printf("\t\t \* \* \*\n");

printf("\t\tA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*C\n\n");

SetConsoleTextAttribute(hStdOut, (WORD)((0 << 4) | 15));

if (key == 0) cout << "\t -> Расчет гипотенузы прямоугольного треугольника по его катетам. " << endl;

else cout << "\t Расчет гипотенузы прямоугольного треугольника по его катетам." << endl;

if (key == 1) cout << "\t -> Поиск периметра фигуры ABCD по заданным сторонам AB, AC, DC.\n" << endl;

else cout << "\t Поиск периметра фигуры ABCD по заданным сторонам AB, AC, DC.\n" << endl;

SetConsoleTextAttribute(hStdOut, (WORD)((0 << 4) | 8));

printf("\t2. Дано натуральное число N. Выведите слово YES, если число N является\n");

printf("\t точной степенью двойки, или слово NO в противном случае.\n");

printf("\t Операцией возведения в степень пользоваться нельзя!\n");

printf("\t Сделать 2 варианта:\n");

printf("\t а) через рекурсию;\n");

printf("\t б) обычной функцией.\n\n");

SetConsoleTextAttribute(hStdOut, (WORD)((0 << 4) | 15));

if (key == 2) cout << "\t -> Решение при помощи обычной функции." << endl;

else cout << "\t Решение при помощи обычной функции." << endl;

if (key == 3) cout << "\t -> Решение при помощи рекурсии.\n" << endl;

else cout << "\t Решение при помощи рекурсии.\n" << endl;

if (key == 4) cout << "\t -> Выход.\n" << endl;

else cout << "\t Выход.\n" << endl;

code = \_getch();

if (code == 224)

{

code = \_getch();

if (code == 80) key++;

if (code == 72) key--;

}

} while (code != 13);

system("cls");

return key;

}

//-------------------------------------------------------------

float gip1()

{

float ab, bc, ac;

int m1, m2;

while (1)

{

system("cls");

printf("1. Найти периметр фигуры АВСD по заданным сторонам AB,AC,DC. \n");

printf(" Написать функцию для расчета гипотенузы прямоугольного треугольника \n");

printf(" по его катетам. \n");

printf("\tB\*\n");

printf("\t \* \* \*\n");

printf("\t \* \* \*\n");

printf("\t \* \* \*\n");

printf("\t \* \* \*D угол прямой между BA и AC, BC и CD\n");

printf("\t \* \* \*\n");

printf("\tA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*C\n\n");

cout << "\nВВОДИТЬ ТОЛЬКО ЧИСЛОВЫЕ ЗНАЧЕНИЯ: \n" << endl;

cout << "Введите сторону AB : ";

//cin >> ab;

rewind(stdin);

m1 = scanf("%f", &ab);

cout << "Введите сторону AC : ";

//cin >> ac;

m2 = scanf("%f", &ac);

if (m1 != 0 && m2 != 0)

{

bc = sqrt(pow(ab, 2) + pow(ac, 2));

bc\_2 = bc;

return bc\_2;

break;

}

system("cls");

cout << "Введите число!" << endl;

system("pause");

}

}

float gip2()

{

float bd, cd;

cout << "Введите сторону CD : ";

cin >> cd;

bd = sqrt(pow(bc\_2, 2) + pow(cd, 2));

bd\_2 = bd;

return bd\_2;

}

void result\_gip()

{

cout << "\nГипотенуза BC = " << gip1()<<"\n" << endl;

cout << "\nГипотенуза BD = " << gip2() <<"\n"<< endl;

}

void P()

{

float ab, ac, cd, p,bc,bd;

int m1, m2,m3;

while (1)

{

system("cls");

printf("1. Найти периметр фигуры АВСD по заданным сторонам AB,AC,DC. \n");

printf(" Написать функцию для расчета гипотенузы прямоугольного треугольника \n");

printf("t по его катетам. \n");

printf("\tB\*\n");

printf("\t \* \* \*\n");

printf("\t \* \* \*\n");

printf("\t \* \* \*\n");

printf("\t \* \* \*D угол прямой между BA и AC, BC и CD\n");

printf("\t \* \* \*\n");

printf("\tA\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*C\n\n");

cout << "Введите сторону AB : ";

//cin >> ab;

rewind(stdin);

m1 = scanf("%f", &ab);

cout << "Введите сторону AC : ";

//cin >> ac;

m2 = scanf("%f", &ac);

cout << "Введите сторону CD : ";

m3=scanf("%f", &cd);

if (m1 != 0 && m2 != 0 && m3!=0)

{

bc = sqrt(pow(ab, 2) + pow(ac, 2));

bd = sqrt(pow(bc, 2) + pow(cd, 2));

p = ab + cd + bd + ac;

cout << "\nР(периметр) = " << p << endl;

break;

}

system("cls");

cout << "Введите число!" << endl;

system("pause");

}

//cout << "\nР(периметр) = " << p << endl;

}

void task\_1(int i)

{

if (i == 1)

result\_gip();

if (i == 2)

P();

system("pause");

}

//-------------------------------------------------------------

void no\_recursion()

{

int n;

do

{

system("cls");

printf("2. Дано натуральное число N. Выведите слово YES, если число N является\n");

printf(" точной степенью двойки, или слово NO в противном случае.\n");

printf(" Операцией возведения в степень пользоваться нельзя!\n\n");

cout << "Решение при помощи обычной функции" << endl;

cout << "\nВВОДИТЬ ТОЛЬКО ЧИСЛОВЫЕ ЗНАЧЕНИЯ: \n" << endl;

cout << "Введите число (n): ";

rewind(stdin);

scanf("%d", &n);

cout << "\n";

} while (n<1 || n>100000000);

cout << "Ответ: " << ((n & (n - 1)) ? "NO" : "YES") << endl;

}

bool recursion(double n)

{

if (trunc(n) != ceil(n))

return false;

if ((n == 2) || (recursion(n / 2)))

return true;

}

int output\_recursion()

{

double n;

printf("2. Дано натуральное число N. Выведите слово YES, если число N является\n");

printf(" точной степенью двойки, или слово NO в противном случае.\n");

printf(" Операцией возведения в степень пользоваться нельзя!\n\n");

cout << "Решение при помощи рекурсии" << endl;

cout << "\nВВОДИТЬ ТОЛЬКО ЧИСЛОВЫЕ ЗНАЧЕНИЯ: \n" << endl;

cout << "Введите число (n): ";

cin>>n;

if (recursion(n))

cout << "\nОтвет: YES" << endl;

else

cout << "\nОтвет: NO" << endl;

return 0;

}

void task\_2(int i)

{

if (i == 1)

no\_recursion();

if (i == 2)

output\_recursion();

system("pause");

}

//-------------------------------------------------------------

int main()

{

while (1)

{

setlocale(LC\_ALL, "RUS");

int i1 = menu\_main() + 1;

switch (i1)

{

case 1: {task\_1(1); break; }

case 2: {task\_1(2); break; }

case 3: {task\_2(1); break; }

case 4: {task\_2(2); break; }

case 5: {return 0; }

}

}

}