МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп'ютерні інформаційні технології»

**Лабораторна робота №2**

**з дисципліни «Основи програмної інженерії»**

**на тему: «**Модульне програмування. Метод висхідного програмування.**»**

Виконав:

студент гр.ПЗ1911

Сафонов Д.Є.

Прийняла:

Куроп'ятник О. С.

Дніпро, 2020

**Тема.** Модульне програмування. Метод висхідного програмування.

**Мета.** Отримати практичні навички розробки програм за методом висхідного програмування.

# **Постановка задачі згідно загального та індивідуального завдання.**

Розробити програму-калькулятор для обчислення трьох математичних виразів за індивідуальним завданням. Програма повинна мати текстове меню. В програмі використати вказівники на функції.

Вимоги до програми:

− вибір виразу для обчислення за допомогою текстового меню;

− для виклику функції обчислення виразу використати вказівники на функції;

− передача параметрів через вказівник типу void;

− данні для обчислення виразів вводяться з клавіатури;

− ввід даних для обчислення і безпосереднє обчислення реалізувати в окремих функціях;

− контроль вхідних даних. Вимоги до тексту програми аналогічні вимогам з лабораторної роботі № 1.

Рівень 3 (відмінно). Параметри для виклику функції записати в масив

|  |  |
| --- | --- |
| 6 |  |

# **Схема модульної структури програми.**

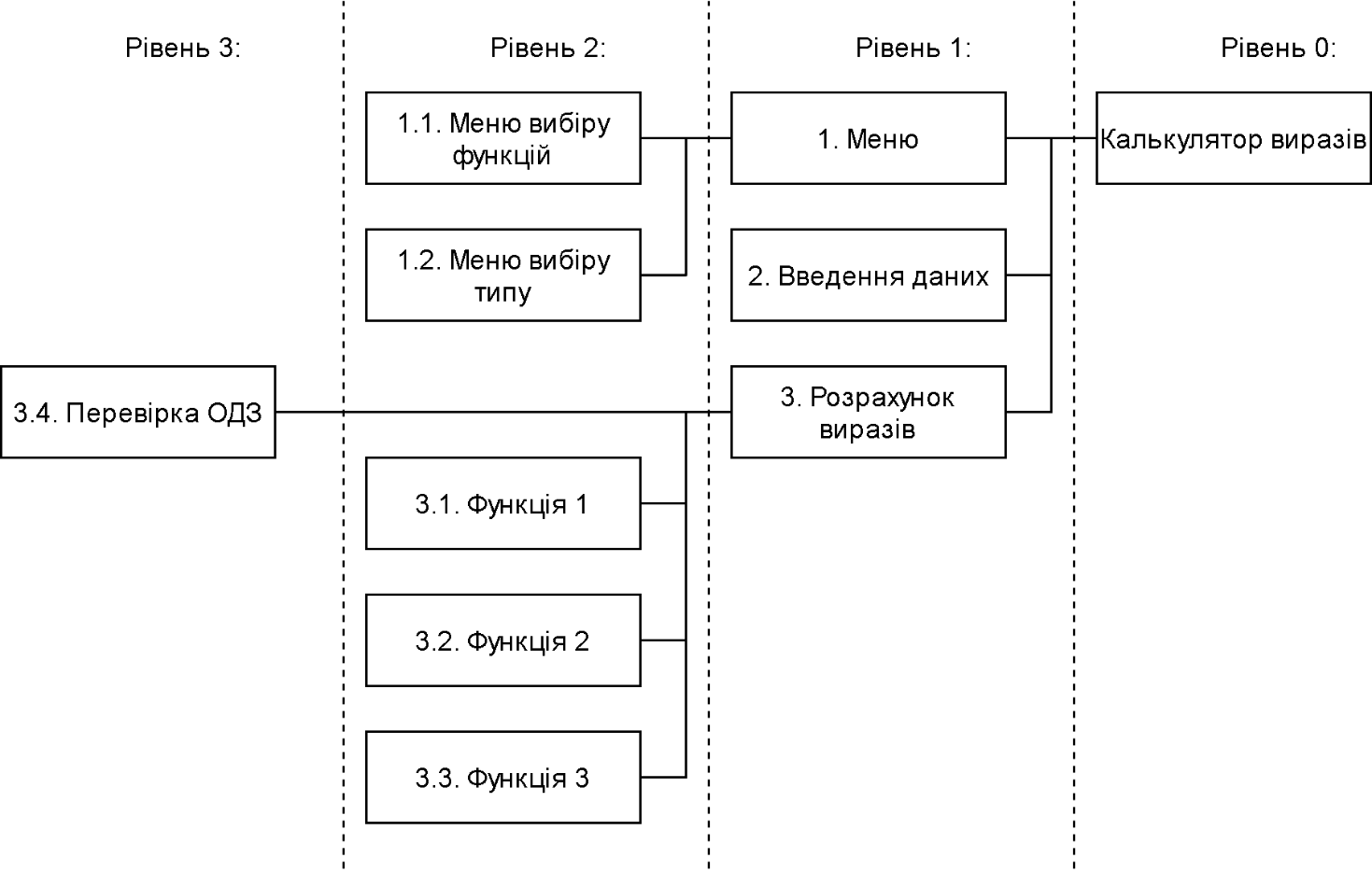


Рисунок 1

# **Протокол розробки програми за методом висхідного програмування.**

По перше функції можна спростити, та зберегти ОДЗ:

|  |  |  |
| --- | --- | --- |
| func | simplified | domain |
|  | -1 |  |
|  |  |  |
|  |  | a>0 |

Для перевірки ОДЗ значення будем повертати в аргумент, а тип функцій буде помилка. Таким чином ми одразу перевіряємо введення даних.

Також розробимо меню за допомогою операторів множинного вибору та індексації масиву.

Файлова структура:

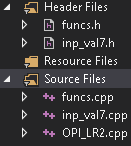


Рисунок 2

# **Текст програми.**

“main.cpp”

#include <iostream>

#include "funcs.h"

#include "inp\_val7.h"

int main() {

func funcs[3]{func1, func2, func3};

bool exit\_ = true;

int choice;

do {

system("cls");

std::cout << "1) 1/(b(abc+a+c))-(1/(a+1/(b+1/c))):(1/(a+1/b))" << std::endl;

std::cout << "a - arg1, b - arg2, c - arg3" << std::endl;

std::cout << "2) (x^(2/m)-9x^(2/n))(x^((1-m)/m)-3x^((1-n)/n))/((x^(1/m)+3x^(1/n))^2-12x^((m+n)/(mn)))" << std::endl;

std::cout << "x - arg1, m - arg2, n - arg3" << std::endl;

std::cout << "3) (a^(1/m)-a^(1/n))^2\*4a^((m+n)/(mn))/((a^(2/m)-a^(2/n))(a^(m+1)+a^((n+1)/n))^(1/m))" << std::endl;

std::cout << "a - arg1, m - arg2, n - arg3" << std::endl;

choice = cinum("choose which function you want to calculate: ", "Wrong input, try again", 1, 1, 1, 3);

void \*arg1 = NULL, \*arg2 = NULL, \*arg3 = NULL, \*result = NULL;//null to avoid error

TYPE type = chooseType();

switch (type) {

case int\_: {

arg1 = new int;

arg2 = new int;

arg3 = new int;

result = new int;

std::cout << "input arg1 = ";

std::cin >> \*static\_cast<int\*>(arg1);

std::cout << "input arg2 = ";

std::cin >> \*static\_cast<int\*>(arg2);

std::cout << "input arg3 = ";

std::cin >> \*static\_cast<int\*>(arg3);

void\* args[5]{ static\_cast<void\*>(&type), arg1, arg2, arg3, result };

bool goodbit = funcs[choice - 1](args);

if (goodbit)

std::cout << "result = " << \*static\_cast<int\*>(result) << std::endl;

else

std::cout << "Bad input" << std::endl;

break;

}

case double\_: {

arg1 = new double;

arg2 = new double;

arg3 = new double;

result = new double;

std::cout << "input arg1 = ";

std::cin >> \*static\_cast<double\*>(arg1);

std::cout << "input arg2 = ";

std::cin >> \*static\_cast<double\*>(arg2);

std::cout << "input arg3 = ";

std::cin >> \*static\_cast<double\*>(arg3);

void\* args[5]{ static\_cast<void\*>(&type), arg1, arg2, arg3, result };

bool goodbit = funcs[choice - 1](args);

if (goodbit)

std::cout << "result = " << \*static\_cast<double\*>(result) << std::endl;

else

std::cout << "Bad input" << std::endl;

break;

}

}

delete arg1;

delete arg2;

delete arg3;

delete result;

char chExit;

std::cout << "Do you want to exit the programm?(Y/N): ";

std::cin >> chExit;

while ((chExit != 'Y' && chExit != 'y' && chExit != 'N' && chExit != 'n') || (std::cin.peek() != '\n')) {//until correct input

while (std::cin.get() != '\n');//clear istream

std::cout << "wrong input, try again." << std::endl;

std::cout << "Do you want to exit the programm?(Y/N): ";

std::cin >> chExit;

}

if (chExit == 'Y' || chExit == 'y')

exit\_ = true;

else

exit\_ = false;

std::cin.get();//enter

} while (!exit\_);

system("pause");

return 0;

}

“inp\_val7.h”

//cinum - numerical only (R)

//char msg\_invite - your message for inviting user to input their number

//char msg\_error - if error input occures

//int m\_ - check if input is multiple of this number

//bool intFlag - =1(only integer input); =0(any input)

//double left - leftmost point in diapasone for input

//double right - rightmost point in diapasone for input

#include <iostream>

#include <windows.h>

#undef max

#include <cmath>

#include <climits>

int cinum(const char\* msg\_invite, const char\* msg\_error, int m\_, bool intFlag, float left, float right);

//input to char array with lenght validation

void val\_char\_lenght(char array[], int ary\_size);

“inp\_val7.cpp”

#include "inp\_val7.h"

using namespace std;

int cinum(const char\* msg\_invite, const char\* msg\_error, int m\_ = 1, bool intFlag = 0, float left = -2147483648.0, float right = 2147483647.0) {

bool i = 0;

float aaa;

cout << msg\_invite << endl;

do {

cin >> aaa;

int iaaa;

iaaa = (int)aaa;

if (cin.fail() || (aaa < left) || (aaa > right) || ((intFlag == 1) && (ceil(aaa) != floor(aaa))) || ((m\_ != 1) && ((iaaa % m\_) != 0))) {

cout << msg\_error << endl;

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

}

else

i = 1;

} while (!i);

return (int)aaa;

}

void val\_char\_lenght(char array[], int ary\_size) {//input to char array with lenght validation

bool correct = false;

do {

std::cin.getline(array, 24);

if ((strlen(array) > 23) || cin.fail()) {

std::cout << "wrong input, try again" << std::endl;

cin.clear();

std::cin.ignore(numeric\_limits<streamsize>::max(), '\n');

}

else

correct = true;

} while (!correct);

}

“funcs.h”

// struct of void\*\* arguments:

// args[0] is TYPE

// args[1]-args[3] is coordinates in following order: a, b, c; or x, m, n; or a, m, n

// args[4] is result box

#ifndef \_\_FUNCS\_H\_\_

#define \_\_FUNCS\_H\_\_

#include "inp\_val7.h"

typedef bool (\*func)(void\*\*);

enum TYPE {int\_, double\_};//make enum class?

bool func1(void\*\* args);

bool func2(void\*\* args);

bool func3(void\*\* args);

TYPE chooseType();

#endif

“funcs.cpp”

// struct of void\*\* arguments:

// args[0] is TYPE

// args[1]-args[3] is coordinates in following order: a, b, c; or x, m, n; or a, m, n

// args[4] is result box

#include "funcs.h"

#include <iostream>

bool func1(void\*\* args) {

TYPE type = \*reinterpret\_cast<TYPE\*>(args[0]);

double a, b, c;

switch (type) {

case int\_:

a = \*static\_cast<int\*>(args[1]);

b = \*static\_cast<int\*>(args[2]);

c = \*static\_cast<int\*>(args[3]);

break;

case double\_:

a = \*static\_cast<double\*>(args[1]);

b = \*static\_cast<double\*>(args[2]);

c = \*static\_cast<double\*>(args[3]);

break;

}

bool goodbit = false;

if ((b != 0) && (a \* b != -1) && (c != 0) && (a \* b \* c + a + c != 0) && (c != -1 / b) && (c != -1 / (a \* b + 1)) && (b \* c != -1)) {

switch (type) {

case int\_:

\*(static\_cast<int\*>(args[4])) = -1;

break;

case double\_:

\*(static\_cast<double\*>(args[4])) = -1;

break;

}

goodbit = true;

}

return goodbit;

}

bool func2(void\*\* args) {

TYPE type = \*reinterpret\_cast<TYPE\*>(args[0]);

double x, m, n;

switch (type) {

case int\_:

x = \*static\_cast<int\*>(args[1]);

m = \*static\_cast<int\*>(args[2]);

n = \*static\_cast<int\*>(args[3]);

break;

case double\_:

x = \*static\_cast<double\*>(args[1]);

m = \*static\_cast<double\*>(args[2]);

n = \*static\_cast<double\*>(args[3]);

break;

}

bool goodbit = false;

if (x > 0 && m != 0 && n != 0) {

double result = (pow(x, 1 / m) + 3 \* pow(x, 1 / n)) / x;

switch (type) {

case int\_:

\*(static\_cast<int\*>(args[4])) = result;

break;

case double\_:

\*(static\_cast<double\*>(args[4])) = result;

break;

}

goodbit = true;

}

return goodbit;

}

bool func3(void\*\* args) {

TYPE type = \*reinterpret\_cast<TYPE\*>(args[0]);

double a, m, n;

switch (type) {

case int\_:

a = \*static\_cast<int\*>(args[1]);

m = \*static\_cast<int\*>(args[2]);

n = \*static\_cast<int\*>(args[3]);

break;

case double\_:

a = \*static\_cast<double\*>(args[1]);

m = \*static\_cast<double\*>(args[2]);

n = \*static\_cast<double\*>(args[3]);

break;

}

bool goodbit = false;

if (a > 0 && m != 0 && n != 0) {

double result = 4 \* pow(a, (m + n) / (m \* n)) \* (pow(a, 1 / m) - pow(a, 1 / n));

result /= (pow(a, 1 / m) + pow(a, 1 / n)) \* pow(pow(a, m + 1) + pow(a, (n + 1) / n), 1 / m);

switch (type) {

case int\_:

\*(static\_cast<int\*>(args[4])) = result;

break;

case double\_:

\*(static\_cast<double\*>(args[4])) = result;

break;

}

goodbit = true;

}

return goodbit;

}

TYPE chooseType() {

TYPE type\_;

std::cout << "1. int" << std::endl;

std::cout << "2. double" << std::endl;

type\_ = static\_cast<TYPE>(cinum("choose type to which result and args are casted: ", "Wrong input, try again", 1, 1, 1, 2) - 1);

return type\_;

}

# **Аналіз результатів та висновки щодо розробки програми за методом висхідного програмування.**

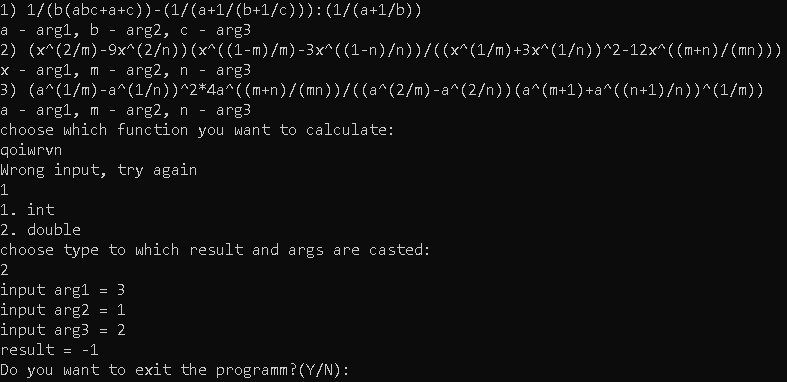


Рисунок 3(Перевірка меню та валідації вхідних даних)

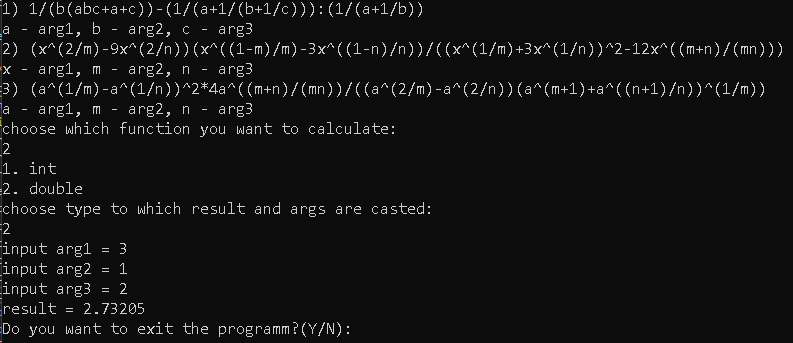


Рисунок 4(Перевірка типів та меню)

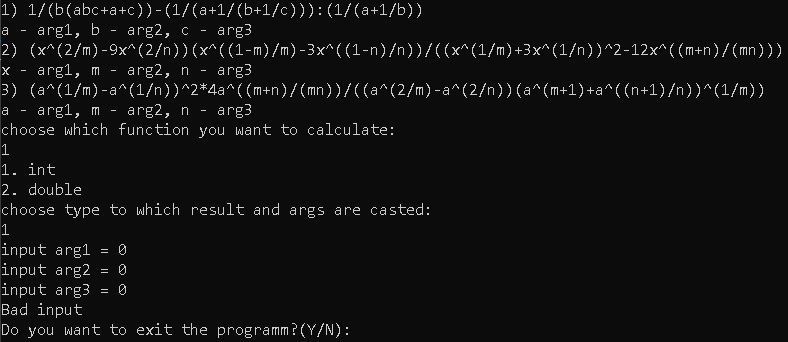


Рисунок 5(Перевірка ОДЗ)

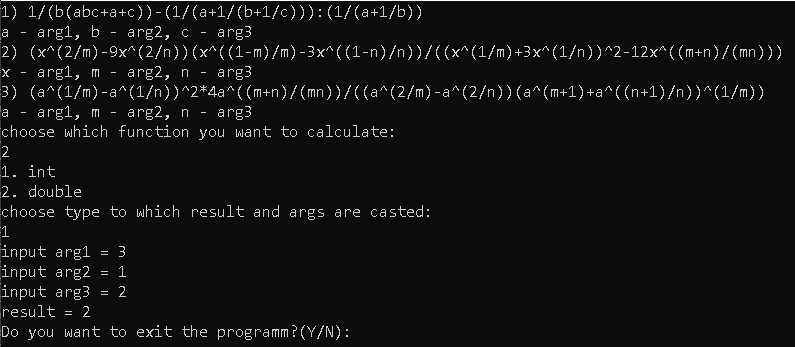


Рисунок 6(Перевірка типів та меню)

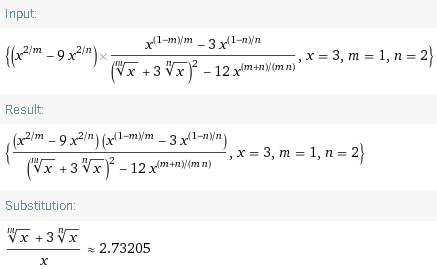


Рисунок 7

Результати відповідають очікуванням, програма працює коректно.

Метод висхідного програмування полягає в розробці спочатку елементарних функцій, а потім все більш складних. Цей метод має декілька переваг. Мабуть найбільша з них полягає в тому, що усі розроблені функції тестуються у процесі розробки і можуть бути одразу використані в проекті.

Але є й недоліки, для цього методу треба ретельно спроектувати програму, перед тим, як починати програмувати.