Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

Звіт

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 8

з навчальної дисципліни

“Базові методології та технології програмування”

ПРОГРАМНА РЕАЛІЗАЦІЯ ОБРОБЛЕННЯ МАСИВІВ ДАНИХ ТА СИМВОЛЬНОЇ ІНФОРМАЦІЇ

ЗАВДАННЯ ВИДАВ

доцент кафедри кібербезпеки та програмного забезпечення

Доренський О. П.

[https://github.com/odorenskyi/](https://github.com/odorenskyi/Dmytro-Parkhomenko-KB18)

ВИКОНАВ

студент академічної групи КН-23

Чубенко І. К.

ПЕРЕВІРИВ

ст. викладач кафедри кібербезпеки   
та програмного забезпечення

Дрєєва Г. М.

Кропивницький 2024

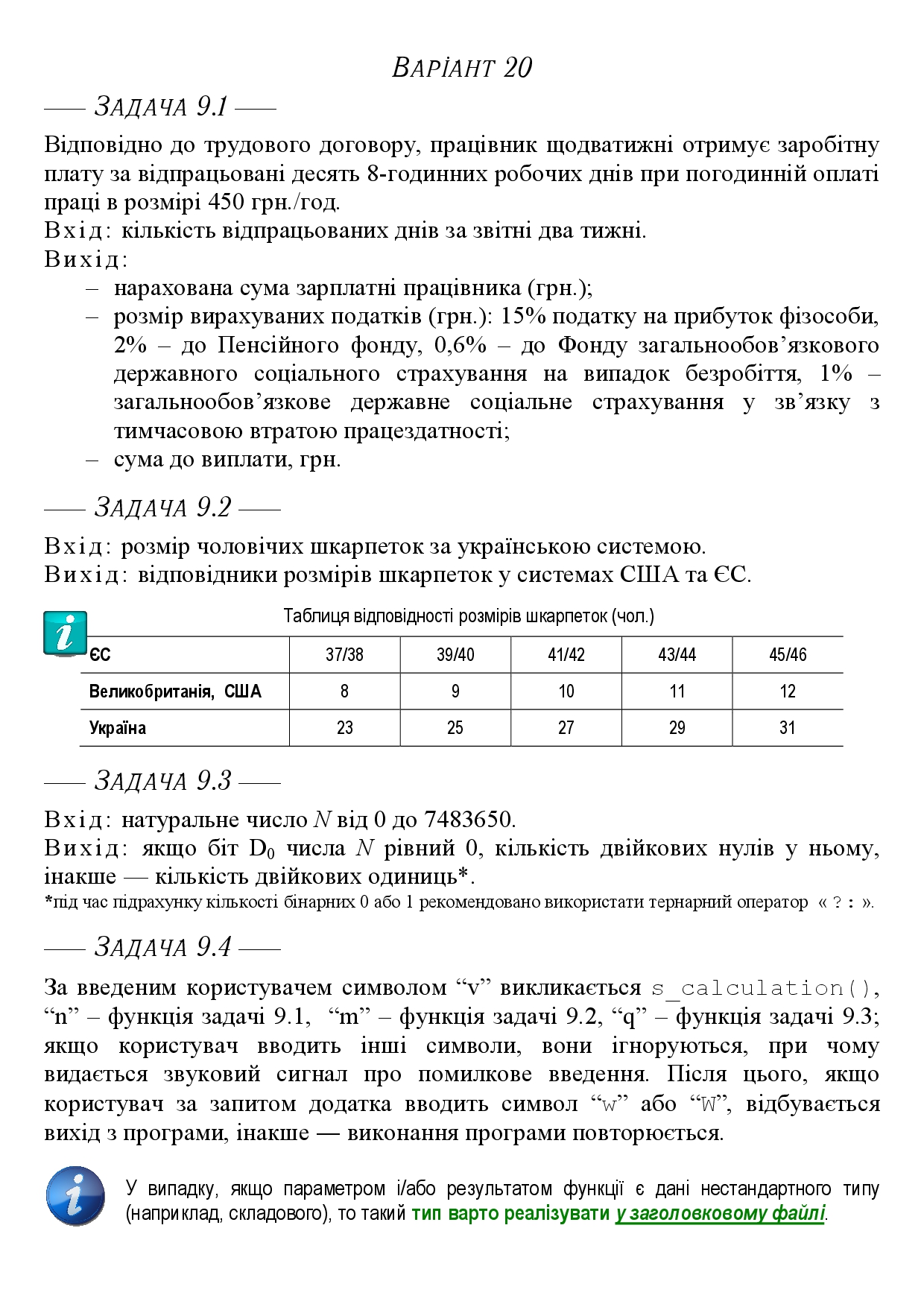
**ТЕМА: РЕАЛІЗАЦІЯ ПРОГРАМНИХ МОДУЛІВ РОЗГАЛУЖЕНИХ ТА ІТЕРАЦІЙНИХ ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ**

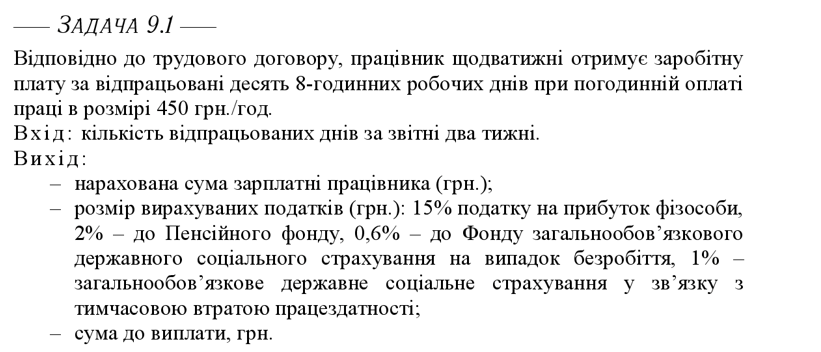
**МЕТА:**полягає у набутті ґрунтовних вмінь і практичних навичок реалізації технології модульного програмування, застосування операторів С++ арифметичних,логічних, побітових операцій, умови, циклів та вибору під час розроблення статичних бібліотек, заголовкових файлів та програмних засобів у кросплатформовому середовищі Code::Blocks.

**ЗАВДАННЯ:**

1. Реалізувати функції розв’язування задач 9.1–9.3 як складових статичної бібліотеки libModulesПрізвище.а (проект ModulesПрізвище, створений під час виконання лабораторної роботи №8).

2. Реалізувати програмне забезпечення розв’язування задачі 9.4 на основі функцій статичної бібліотеки libModulesПрізвище.а.





Вхід: Кількість відпрацьованих днів протягом двох тижнів.

Вивід:

Нарахована сума зарплати працівника (в гривнях).

Розмір вирахованих податків (в гривнях):

15% податку на прибуток фізособи,

2% до Пенсійного фонду,

0,6% до Фонду загальнообов'язкового державного соціального страхування на випадок безробіття,

1% - загальнообов'язкове державне соціальне страхування у зв'язку з тимчасовою втратою працездатності.

Сума до виплати (в гривнях).

Лістинг

void calculateSalary(int day) {

double zar, p1, p2, p3, p4;

double sum;

int oplata = 450;

zar = day \* oplata;

cout << "Зарплатня: " << zar << endl;

p1 = 15.0 / 100 \* zar;

cout << "Податок 15 %: " << p1 << endl;

p2 = 2.0 / 100 \* zar;

cout << "Податок 2 % -: " << p2 << endl;

p3 = 0.6 / 100 \* zar;

cout << "Податок 0,6 % -: " << p3 << endl;

p4 = 1.0 / 100 \* zar;

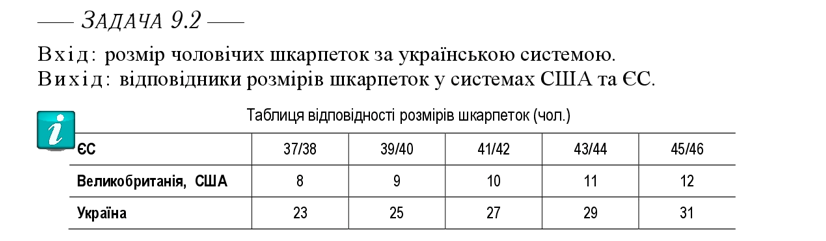
cout << "Податок 1 %: " << p4 << endl;

sum = p1 + p2 + p3 + p4;

cout << "Сума до виплати: " << zar - sum << endl;

|  |  |  |  |
| --- | --- | --- | --- |
| ID TS | Вхід | Вихід | Результат тестування /  Test Result |
| TC\_01 | Введіть кількість відпрацьованих днів:34 | Зарплатня: 15300  Податок 15 %: 2295  Податок 2 % -: 306  Податок 0,6 % -: 91.8  Податок 1 %: 153  Сума до виплати: 12454.2 | passed |
| TC\_02 | Введіть кількість відпрацьованих днів: 43 | Зарплатня: 19350  Податок 15 %: 2902.5  Податок 2 % -: 387  Податок 0,6 % -: 116.1  Податок 1 %: 193.5  Сума до виплати: 15750.9 | passed |
| TC\_03 | Введіть кількість відпрацьованих днів: 54 | Зарплатня: 24300  Податок 15 %: 3645  Податок 2 % -: 486  Податок 0,6 % -: 145.8  Податок 1 %: 243  Сума до виплати: 19780.2 | passed |
| TC\_04 | Введіть кількість відпрацьованих днів: 31 | Зарплатня: 13950  Податок 15 %: 2092.5  Податок 2 % -: 279  Податок 0,6 % -: 83.7  Податок 1 %: 139.5  Сума до виплати: 11355.3 | passed |
| TC\_05 | Введіть кількість відпрацьованих днів: 21 | Зарплатня: 9450  Податок 15 %: 1417.5  Податок 2 % -: 189  Податок 0,6 % -: 56.7  Податок 1 %: 94.5  Сума до виплати: 7692.3 | passed |
| TC\_06 | Введіть кількість відпрацьованих днів: 11 | Зарплатня: 4950  Податок 15 %: 742.5  Податок 2 % -: 99  Податок 0,6 % -: 29.7  Податок 1 %: 49.5  Сума до виплати: 4029.3 | passed |
| TC\_07 | Введіть кількість відпрацьованих днів: 23 | Зарплатня: 10350  Податок 15 %: 1552.5  Податок 2 % -: 207  Податок 0,6 % -: 62.1  Податок 1 %: 103.5  Сума до виплати: 8424.9 | passed |

|  |  |  |  |
| --- | --- | --- | --- |
| TC\_08 | Введіть кількість відпрацьованих днів: 43 | Зарплатня: 19350  Податок 15 %: 2902.5  Податок 2 % -: 387  Податок 0,6 % -: 116.1  Податок 1 %: 193.5  Сума до виплати: 15750.9 | passed |
| TC\_09 | Введіть кількість відпрацьованих днів: 57 | Зарплатня: 25650  Податок 15 %: 3847.5  Податок 2 % -: 513  Податок 0,6 % -: 153.9  Податок 1 %: 256.5  Сума до виплати: 20879.1 | passed |
| TC\_10 | Введіть кількість відпрацьованих днів: 12 | Зарплатня: 5400  Податок 15 %: 810  Податок 2 % -: 108  Податок 0,6 % -: 32.4  Податок 1 %: 54  Сума до виплати: 4395.6 | passed |



Задача 9.2 Програма очікує введення користувачем розміру чоловічих шкарпеток в українській системі розмірів.

Після отримання введеного значення, програма перевіряє, чи відповідає воно одному з таких умов:

Якщо розмір дорівнює 23 або 24, програма вважає, що це розмір 37 або 38 в Європейській системі та розмір 8 в системі США.

Якщо розмір дорівнює 25 або 26, програма вважає, що це розмір 39 або 40 в Європейській системі та розмір 9 в системі США.

Якщо розмір дорівнює 27 або 28, програма вважає, що це розмір 41 або 42 в Європейській системі та розмір 10 в системі США.

Якщо розмір дорівнює 29 або 30, програма вважає, що це розмір 43 або 44 в Європейській системі та розмір 11 в системі США.

Якщо розмір дорівнює 31, програма вважає, що це розмір 45 або 46 в Європейській системі та розмір 12 в системі США.

Якщо жодна з перелічених умов не виконується, програма виводить повідомлення про те, що введений розмір не відповідає жодному варіанту.

Після виконання порівняння програма завершує свою роботу.

Лістинг

Ввести розмір

string convertSockSize(int rozm) {

if (rozm == 23 || rozm == 24) {

return "В ЄС 37/38\nВ США 8\n";

}

else if (rozm == 25 || rozm == 26) {

return "В ЄС 39/40\nВ США 9\n";

}

else if (rozm == 27 || rozm == 28) {

return "В ЄС 41/42\nВ США 10\n";

}

else if (rozm == 29 || rozm == 30) {

return "В ЄС 43/44\nВ США 11\n";

}

else if (rozm == 31) {

return "В ЄС 45/46\nВ США 12\n";

}

else {

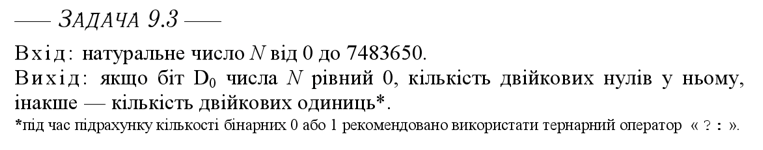
return "Розмір не відповідає жодному варіанту.\n";

}

}

|  |  |  |  |
| --- | --- | --- | --- |
| ID TS | Вхід | Вихід | Результат тестування /  Test Result |
| TC\_01 | Введіть розмір чоловічих шкарпеток за українською системою: 23 | В ЄС 37/38  В США 8 | passed |
| TC\_02 | Введіть розмір чоловічих шкарпеток за українською системою: 24 | В ЄС 37/38  В США 8 | passed |
| TC\_03 | Введіть розмір чоловічих шкарпеток за українською системою: 25 | В ЄС 39/40  В США 9 | passed |
| TC\_04 | Введіть розмір чоловічих шкарпеток за українською системою: 26 | В ЄС 39/40  В США 9 | passed |
| TC\_05 | Введіть розмір чоловічих шкарпеток за українською системою: 27 | В ЄС 41/42  В США 10 | passed |
| TC\_06 | Введіть розмір чоловічих шкарпеток за українською системою: 28 | В ЄС 41/42  В США 10 | passed |
| TC\_07 | Введіть розмір чоловічих шкарпеток за українською системою: 29 | В ЄС 43/44  В США 11 | passed |

|  |  |  |  |
| --- | --- | --- | --- |
| TC\_08 | Введіть розмір чоловічих шкарпеток за українською системою: 30 | В ЄС 43/44  В США 11 | passed |
| TC\_09 | Введіть розмір чоловічих шкарпеток за українською системою: 31 | В ЄС 45/46  В США 12 | passed |
| TC\_10 | Введіть розмір чоловічих шкарпеток за українською системою: 32 | Розмір не відповідає жодному варіанту. | passed |



1.Запросити в користувача ввести натуральне число N у діапазоні від 0 до 7483650.

2.Обчислити кількість нулів та одиниць у бінарному представленні числа N.

3.Вивести результат: якщо введене число N рівне 0, вивести кількість нулів; в іншому випадку, вивести кількість одиниць.

int countOnesAndZeros( int N) {

int zero\_count = 0;

int result=0;

int one\_count = 0;

for (int i = 0; i < 32; ++i) {

if (N & (1 << i)) {

++one\_count;

}

else {

++zero\_count;

}

}

(N == 0) ? result=zero\_count : result=one\_count;

return result;

}

|  |  |  |  |
| --- | --- | --- | --- |
| ID TS | Вхід | Вихід | Результат тестування /  Test Result |
| TC\_01 | Enter a natural number N from 0 to 7483650: 5 | Result for number 5: 2 | passed |
| TC\_02 | Enter a natural number N from 0 to 7483650: 36 | Result for number 36: 2 | passed |
| TC\_03 | Enter a natural number N from 0 to 7483650: 62 | Result for number 62: 5 | passed |
| TC\_04 | Enter a natural number N from 0 to 7483650: 68 | Result for number 68: 2 | passed |
| TC\_05 | Enter a natural number N from 0 to 7483650: 76 | Result for number 76: 3 | passed |
| TC\_06 | Enter a natural number N from 0 to 7483650: 83 | Result for number 83: 4 | passed |
| TC\_07 | Enter a natural number N from 0 to 7483650: 91 | Result for number 91: 5 | passed |

|  |  |  |  |
| --- | --- | --- | --- |
| TC\_08 | Enter a natural number N from 0 to 7483650: 0 | Result for number 0: 32 | passed |
| TC\_09 | Enter a natural number N from 0 to 7483650: 111 | Result for number 111: 6 | passed |
| TC\_10 | Enter a natural number N from 0 to 7483650: 13 | Result for number 13: 3 | passed |

Лістинг TestDriver

#include <iostream>

#include <windows.h>

#include"ModulesChubenko.h"

using namespace std;

void testThirdTask(int test\_case,int N, unsigned int expected) {

cout << "Тест-кейс №" << test\_case << endl;

cout << "Вхідні дані:\nN: " << N << endl;

unsigned int result = countOnesAndZeros(N);

cout << "Результат: " << result << endl;

if (result == expected) {

cout << "Статус тест-кейса: passed" << endl;

} else {

cout << "Статус тест-кейса: failed" << endl;

}

cout << endl;

}

int main()

{

testThirdTask(1,245, 2);

system("chcp 65001 & cls");

cout << "\n\n9.1\n" << endl;

calculateSalary(34);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(43);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(54);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(31);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(21);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(11);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(23);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(43);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(57);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

calculateSalary(12);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << "\n\n9.2\n" << endl;

cout << convertSockSize(23) <<endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(24) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(25) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(26) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(27) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(28) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(29) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(30) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(31) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << convertSockSize(32) << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

cout << "\n\n9.3\n" << endl;

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(1, 5, 2);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(2, 36, 2);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(3, 62, 5);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(4, 68, 2);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(5, 76, 3);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(6, 83, 4);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(7, 91, 5);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(8, 0, 32);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

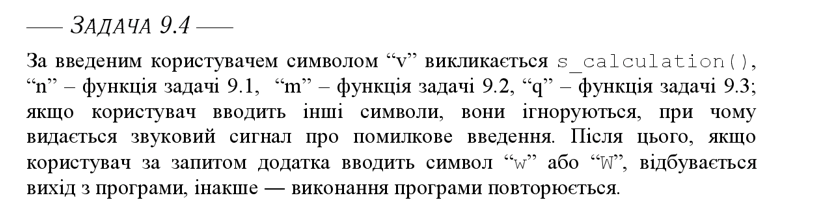
testThirdTask(9, 111, 6);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

testThirdTask(10,13, 3);

cout<<"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}



Алгоритм

1.Вести x

2.Якщо x=”v” то визвати функцію s\_calculation

3. Якщо x=”n” то визвати функцію задачі 9.1

4. Якщо x=”m” то визвати функцію задачі 9.2

5.Якщо x=”q” то визвати функцію задачі 9.3

6.Якщо x != ”v” && x != ”n” && x != ”m” && x != ”q” то видати звуковий сигнал и повернуться на крок 1.

7.Вести х

8.Якщо x=”w” || x=”W” || x=”p” то повернуться на крок 2.

9.Кінець

#include <iostream>

#include <conio.h>

#include "ModulesChubenko.h"

using namespace std;

int main() {

system("chcp 1251 & cls");

char function\_choice;

do {

printCopyright();

cout << "Оберіть функцію, що бажаєте виконати (\"v\", \"n\", \"m\", \"q\"): ";

cin >> function\_choice;

cout << endl;

switch (function\_choice) {

case 'v': {

int x;

cout << "x: ";

cin >> x;

int y;

cout << "y: ";

cin >> y;

int z;

cout << "z: ";

cin >> z;

cout << "Результат функції s\_calculation(): ";

s\_calculation(x, y, z);

break;

}

case 'n': {

int day;

cout << "day: ";

cin >> day;

cout << "Результат функції 9.1: ";

calculateSalary(day);

break;

}

case 'm': {

int rozm;

cout << "rozm: ";

cin >> rozm;

cout << "Розмір носка: " << convertSockSize(rozm);

break;

}

case 'q': {

int N;

cout << "N: ";

cin >> N;

cout << "Результат функції 9.3: " << countOnesAndZeros(N)<<endl;

break;

}

default: {

cout << "Вибрана функція не підтримується." << endl;

break;

}

}

cout << "\nНатисніть \"w\" або \"W\" або \"p\" для завершення виконання програми інакше для повтору виконання програми";

function\_choice = getch();

} while (function\_choice != 'w' && function\_choice != 'W' && function\_choice != 'p');

return 0;

}

Результати виконання тестового драйвера

|  |  |  |  |
| --- | --- | --- | --- |
| Ід-р тест-кейса / Test Case ID | Дії (кроки) /  Action (Test Steps) | Очікуваний  результат /  Expected Result | Результат тестування /  Test Result |
| TC\_01 | 1.Вести v  2.Вести 8  3.Вести 2  4.Вести 16 | S: 5.383  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми | Passed |
| TC\_02 | 1.Вести v  2.Вести 32  3.Вести 43  4.Вести 35 | Результат обчислень невизначений або недійсний.  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми | Passed |
| TC\_03 | 1.Вести n  2.Вести 31 | Результат функції 9.1 Зарплатня: 13950.000  Податок 15 %: 2092.500  Податок 2 % -: 279.000  Податок 0,6 % -: 83.700  Податок 1 %: 139.500  Сума до виплати: 11355.300  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми | Passed |
| TC\_04 | 1.Вести n  2.вести 77 | Результат функції 9.1: Зарплатня: 34650.000  Податок 15 %: 5197.500  Податок 2 % -: 693.000  Податок 0,6 % -: 207.900  Податок 1 %: 346.500  Сума до виплати: 28205.100  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми | Passed |
| TC\_05 | 1.Вести m  2.Вести 26 | Результат функції 9.1: В ЄС 39/40 В США 9  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми | Passed |
| TC\_06 | 1.Вести m  2.вести 55 | Розмір не відповідає жодному варіанту. | Passed |
| TC\_07 | 1.Вести q  2.Вести 456 | Результат функції 9.3: 4  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми | Passed |
| TC\_08 | 1.Вести q  2.вести 1234567 | Результат функції 9.3: 11  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми | Passed |
| TC\_09 | 1.Вести q  2.вести 17683253532768532784 | Результат функції 9.3: 31  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми повтору виконання програми  -----------------------------  | Chubenko Ihor, CNTU, https://github.com/IgorChubenkoKN-23 |  | Chubenko Ihor, CNTU, https://github.com/IgorChubenkoKN-23 |  ------------ © All rights reserved ------------ Оберіть функцію, що бажаєте виконати ("v", "n", "m", "q")  Результат функції 9.1: Error enter within [1,100]  Натисніть "w" або "W" або "p" для завершення виконання програми інакше для повтору виконання програми повтору виконання програми | Passed |
| TC\_10 | 1.Натиснути | Виконання програми завершено | Passed |

Виконуючи дану лабораторну роботу, я отримав значні знання та навички у використанні модульного програмування засобами С++, управління операторами арифметичних, логічних та побітових операцій, а також використання умов, циклів та вибору в середовищі Code::Blocks.

Лабораторна робота №9 мала особливість - необхідність використання бібліотеки, створеної під час 8 лабораторної роботи, та її доповнення новими модулями. Завдяки цьому ми змогли ефективно виконати поставлені завдання, які передбачали роботу з цією бібліотекою.

У задачах 9.1, 9.2 та 9.3 ми також створили модулі для подальшого використання. А от у завданні 9.4 ми реалізували власний програмний застосунок.

Перш ніж реалізувати завдання, я виконав аналіз вимог, спроектував архітектуру та докладно розробив програмні модулі для вирішення задач 9.1–9.3. Я також підготував контрольні приклади для модульного тестування функцій.

Після цього я відкрив проект статичної бібліотеки у Code::Blocks, який був створений у 8 лабораторній роботі. За вже розробленими артефактами виконав реалізацію функцій для завдань 9.1, 9.2 та 9.3. Після компіляції проекту статичної бібліотеки я створив консольний застосунок для тестування розроблених функцій.

Завершивши модульне тестування, я приступив до аналізу та постановки завдання 9.4. Після ретельного проектування програмного забезпечення я реалізував його за допомогою функцій статичної бібліотеки та провів системне тестування.

Ця робота була дуже цікавою і корисною, оскільки ми використовували і покращували раніше створену бібліотеку. Реалізація завдань через модулі виявилася дуже зручною, особливо для подальшого використання.

**Контрольні запитання:**

1. Яким оператором С/С++ можливо повноцінно замінити тернарний оператор? Відповідь обґрунтуйте й доведіть експериментально.

if (condition) {

result = value\_if\_true;

} else {

result = value\_if\_false;

}

1. Що в програмуванні розуміють під пріоритетом виконання операцій та асоціативністю?

**Пріоритет виконання операцій:** Це числове значення, що вказує на пріоритет або пріоритетність операцій у виразі. Операції з вищим пріоритетом виконуються першими. Наприклад, у виразі 5 + 3 \* 2, множення має вищий пріоритет ніж додавання, тому операція 3 \* 2 буде виконана першою, а потім результат буде доданий до числа 5.

**Асоціативність:** Це властивість операції вказувати, у якому напрямку виконується операція, коли у виразі є кілька операцій з однаковим пріоритетом. Наприклад, у виразі 5 - 3 - 1, оператор віднімання є лівоасоціативним, тобто виконується зліва направо. Спочатку відбудеться операція 5 - 3, потім від результату відніметься число 1.

1. Яку область видимості мають змінні, оголошені в тілі циклу або умови (вибору)? Відповідь обґрунтуйте та доведіть експериментально.

#include <iostream>

int main() {

for (int i = 0; i < 3; ++i) {

int x = i \* 2;

std::cout << "Inside loop: x = " << x << std::endl;

}

// Спроба використати змінну x поза циклом

std::cout << "Outside loop: x = " << x << std::endl; // Помилка компіляції

return 0;

}

1. Якою є асоціативність операцій арифметичних, логічних, логічних порозрядних, інкреметна, декремента, тернарної та порівняння в мові програмування С/С++?

Арифметичні операції: асоціативність арифметичних операцій (додавання, віднімання, множення, ділення) є лівоасоціативною. Це означає, що операції виконуються зліва направо. Наприклад, вираз a - b - c обробляється як (a - b) - c.

Логічні та логічні порозрядні операції: асоціативність логічних та логічних порозрядних операцій (AND, OR, XOR) є лівоасоціативною. Наприклад, вираз a && b && c обробляється як (a && b) && c.

Інкрементна та декрементна операції: асоціативність інкрементних та декрементних операцій (++, --) є правоасоціативною. Наприклад, вираз ++a++ буде спочатку збільшувати a, а потім збільшувати результат.

Тернарна (условна) операція: тернарна операція (записана як condition ? value\_if\_true : value\_if\_false) є правоасоціативною. Тобто, вирази типу a ? b : c ? d : e будуть оцінюватися як a ? b : (c ? d : e).

Операції порівняння: операції порівняння (наприклад, <, >, ==) також є безасоціативними. Це означає, що вирази, які містять більше одного порівняння, потрібно обговорювати чітко, вказуючи порядок дій.

1. Перелічіть випадки, за яких доцільно використовувати тернарний оператор С/С++, й наведіть приклад його запису.

Тернарний оператор у мовах програмування C/C++ (вираз у формі условие ? выражение1 : выражение2) дозволяє зручно замінити конструкцію if-else у випадках, коли потрібно вибрати одне з двох значень залежно від певної умови. Такий підхід може бути доцільним у таких випадках:

Присвоєння значення залежно від умови: Коли необхідно присвоїти змінній одне значення, якщо умова виконується, або інше значення, якщо умова не виконується.

int a = 10;

int b = (a > 5) ? 100 : 200; // Якщо a > 5, b = 100, інакше b = 200

Форматування рядків: Тернарний оператор може бути корисним для форматування рядків залежно від певної умови.

int x = 5;

std::string message = (x > 0) ? "Positive" : "Non-positive";

Умовне виведення: Використання тернарного оператора може бути зручним у функціях, які повертають значення, залежно від умови.

int max(int a, int b) {

return (a > b) ? a : b;

}

1. Яке значення міститиме змінна cnt після виконання наступної інструкції: cnt--; ?

Після виконання інструкції cnt--; змінна cnt буде зменшена на одиницю. Це є скороченою формою інкременту, яка віднімає одиницю від поточного значення змінної cnt.

Таким чином, якщо, наприклад, початкове значення змінної cnt дорівнює 5, після виконання інструкції cnt--; її значення становитиме 4.

1. Чим константна змінна, оголошена за допомогою кваліфікатора типів const, відрізняється від змінної? Сформулюйте правило, коли змінну варто оголошувати саме константною.

Константна змінна, оголошена за допомогою кваліфікатора типів const, відрізняється від звичайної змінної тим, що її значення не може бути змінене після ініціалізації. Тобто, після того як константій змінній присвоєно значення, воно залишається незмінним протягом усього часу життя змінної.

Правило оголошення змінної саме константною полягає в тому, що коли ви бажаєте, щоб значення змінної залишалося постійним під час виконання програми, і ви не плануєте змінювати це значення пізніше в коді. Такі змінні забезпечують безпеку програми та зроблюють її стійкішою до помилок.

1. Яких типів можуть бути операнди логічних операторів С/С++?

Операндами логічних операторів у мовах програмування C/C++ можуть бути значення типу bool, які представляють логічну істину або ложь. Оператори логічних виразів працюють з цими значеннями і повертають результат також у формі bool.

Типи операндів логічних операторів:

bool: Значення типу bool можуть бути операндами логічних операторів. Вони можуть мати значення true або false.

Приклади використання логічних операторів з bool-операндами:

bool a = true;

bool b = false;

bool result\_and = a && b; // Логічне "І" (AND)

bool result\_or = a || b; // Логічне "АБО" (OR)

bool result\_not = !a; // Логічне "НЕ" (NOT)

1. Яке значення міститиме змінна cnt при: bool cnt = !!0; ?

У виразі !!0, спочатку застосовується оператор ! до значення 0, що перетворює його на true, оскільки !0 поверне true, а потім другий оператор ! знову використовується до отриманого значення, перетворюючи його на false, оскільки !!0 повертає протилежне значення логічного true, тобто false.

Отже, після виконання інструкції bool cnt = !!0;, змінна cnt буде містити значення false.

1. Сформулюйте правило запису виразу ініціалізації у циклах з параметром (for) С++.

Вираз ініціалізації може містити одну або кілька змінних: Ви можете ініціалізувати одну або кілька змінних в цьому виразі. Наприклад:

for (int i = 0; i < 10; ++i) {

// код циклу

}

У цьому прикладі int i = 0 - це вираз ініціалізації, де i - змінна, яка ініціалізується значенням 0 перед входженням в цикл.

Змінні можуть бути будь-якого типу, який підтримується мовою C++: Змінні виразу ініціалізації можуть бути будь-якого типу даних, що підтримується мовою C++. Наприклад, це може бути int, double, char або будь-який користувацький тип даних.

Вираз ініціалізації є необов'язковим: Ви можете опустити вираз ініціалізації, якщо немає потреби ініціалізувати змінні перед входженням в цикл. У такому випадку ви можете використовувати змінні, які були оголошені попередньо.

Вираз ініціалізації може містити вирази присвоювання та інші вирази: Ви можете використовувати будь-які вирази, які валідні для присвоєння значення змінним.

Змінні, що ініціалізуються, можуть використовуватися в умові та інкременті циклу: Змінні, які ініціалізуються в цьому виразі, можуть бути використані умові та інкременті циклу.