**ДОДАТОК А**

(Вихідний код ModulesStriuk)

**… / ModulesStriuk / main.cpp**

#include <iostream>

#include <cmath>

#include "ModulesStriuk.h"

using namespace std;

double s\_calculation(float x, float y, float z)

{

const double PI = 3.14159;

return log(x - y) + sqrt((PI \* pow(x, 2)) / x + (z / (2 \* pow(y, 2))));

}

money salaryAndTaxesOutput(int workingHours[5])

{

cout << "========================================================================" << endl;

/// Перевірка на правильність вказання часу

for (int i = 0; i < 5; i++) {

if (workingHours[i] < 0) {

cout << endl << "ПОМИЛКА: від\'ємне значення зарплати ! (автоматично визначається як 0)" << endl << endl;

workingHours[i] = 0;

}

}

cout << "ПН: " << workingHours[0] << endl

<< "ВТ: " << workingHours[1] << endl

<< "СР: " << workingHours[2] << endl

<< "ЧТ: " << workingHours[3] << endl

<< "ПТ: " << workingHours[4] << endl << endl;

/// Нарахування ЗП згідно з відпрацьованих годин

int fullSalary = 0;

for (int i = 0; i < 5; i++) {

if (workingHours[i] <= 0) { continue; }

else if (workingHours[i] <= 8) { fullSalary += workingHours[i] \* 40; }

else { fullSalary += workingHours[i] \* 40 + 300; }

}

/// Визначення податків та ЗП у кінечному результаті

money m;

m.salary = fullSalary;

m.fizosoba = fullSalary \* 0.15;

m.pensiya = fullSalary \* 0.02;

m.bezrob = fullSalary \* 0.006;

m.pratsa = fullSalary \* 0.01;

m.completeSalary = fullSalary - (fullSalary \* 0.186);

cout << "Нарахована сума зарплатні: "

<< m.salary << " грн" << endl;

cout << "Податок на прибуток фізособи (15%): "

<< m.fizosoba << " грн" << endl

<< "Податок до пенсійного фонду (2%): "

<< m.pensiya << " грн" << endl

<< "Податок до фонду страхування на випадок безробіття (0.6%): "

<< m.bezrob << " грн" << endl

<< "Податок на соціально страхування у випадку втрати працезатності (1%): "

<< m.pratsa << " грн" << endl;

cout << "Сумма до видачі: " << m.completeSalary << " грн" << endl << endl << endl;

return m;

}

socks socksSizeStandart(unsigned char sSize)

{

/// Заверешення функції при недопустимих вхідних даних

if (sSize != 23 && sSize != 25 && sSize != 27 && sSize != 29 && sSize != 31) {

cout << "ПОМИЛКА: Розмір шкарпеток не відповідає стандарту !" << endl;

return {-1, -1, -1, -1};

}

/// Виведення аналогів розміру (для спрощення коду, за знайденною закономірністю, використовуються формули)

socks s;

s.ukrSize = sSize;

s.usSize = (sSize / 2) - 2;

s.esSize1 = sSize + 14;

s.esSize2 = sSize + 15;

cout << "Розмір шкарпеток за українською системою: " << s.ukrSize << endl;

cout << "Розмір шкарпеток за системою США: " << s.usSize << endl;

cout << "Розмір шкарпеток за системою ЄС: " << s.esSize1 << "//" << s.esSize2 << endl;

return s;

}

int numberByteManipulation(int number)

{

/// Завершення функції, при невідповідних до заданого діапазону чисел

if (number < 0 || number > 7483650) {

cout << "ПОМИЛКА: число в недопустимому діапазоні !" << endl;

return -1;

}

int reverseNumber = 0;

int bitCount = 0;

/// Інвертування числа (біти, за замовчуванням, рахуються з кінця числа)

for (; number; number /= 2) {

reverseNumber = reverseNumber \* 2 + number % 2;

}

number = reverseNumber;

/// Додавання до кількості бітів результату логічного виразу (біт не має бути рівним сьомому)

int saveIndex = 0;

for (int i = 0; i <= 31; i++) {

if (number & (1 << i)) { saveIndex = i + 1; }

bitCount += ((number & (1 << i)) != ((number >> 7) & 1));

}

if ((number >> 7) & 1) { bitCount = bitCount - (31 - saveIndex); }

switch ((number >> 7) & 1) {

case 0: cout << "Кількість двійкових 1 = " << bitCount << endl; break;

case 1: cout << "Кількість двійкових 0 = " << bitCount << endl; break;

}

return bitCount;

}

**… / ModulesStriuk / ModulesStriuk.h**

#ifndef MODULESSTRIUK\_H\_INCLUDED

#define MODULESSTRIUK\_H\_INCLUDED

struct money {

float salary;

float fizosoba;

float pensiya;

float bezrob;

float pratsa;

float completeSalary;

};

struct socks {

int ukrSize;

float usSize;

int esSize1;

int esSize2;

};

double s\_calculation(float x, float y, float z);

money salaryAndTaxesOutput(int workingHours[5]);

socks socksSizeStandart(unsigned char ukrSoksSize);

int numberByteManipulation(int number);

#endif // MODULESSTRIUK\_H\_INCLUDED