// ConsoleApplication23.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

using namespace std;

================= ЗНАЧЕННЯ ЗА ЗАМОВЧУВАННЯМ (ПРИКЛАД) =============

//int getRandomValue(int minValue=0, int maxValue=\_I32\_MAX)

//{

// return minValue + rand() % (maxValue-minValue+1);

//}

================ ПЕРЕВАНТАЖЕННЯ ФУНКЦІЙ (ПРИКЛАД) =================

//int getMax(int a, int b)

//{

// return a > b ? a : b;

//}

//int getMax(int a, int b, int c)

//{

// return getMax( getMax(a,b) , c);

//}

//int getMax(int a, int b, int c, int d)

//{

// return getMax(getMax(a, b), getMax(c,d));

//}

======================= ПЕРЕДАЧА ДОВІЛЬНОЇ КІЛЬКОСТІ ПАРАМЕТРІВ =========

//---------------Вимагає створення масиву

//int getMax(int\* numbers, int n)

//{

// int m = numbers[0];

// for (int i = 1; i < n; i++)

// {

// if (m>numbers[i])

// {

// m = numbers[i];

// }

// }

// return m;

//}

////--------------- Передача довільної кількості елементів

//int getMax(int n, int firstElement,...)

//{

// int\* numbers = &firstElement; //&firstElement - операція взяття адреси параметра firstElement

// int m = numbers[0];

// for (int i = 1; i < n; i++)

// {

// if (m>numbers[i])

// {

// m = numbers[i];

// }

// }

// return m;

//}

========================= ФУНКЦІЯ, З ПАРАМЕТРАМИ ДОВІЛЬНОГО ТИПУ ==

-------------- ПІДХІД 1 ВИКОРИСТОВУВАТИ ПЕРЕВАНТАЖЕНІ ФУНКЦІЇ ---------

//int getMax(int a, int b)

//{

// return a > b ? a : b;

//}

//double getMax(double a, double b)

//{

// return a > b ? a : b;

//}

//char getMax(char a, char b)

//{

// return a > b ? a : b;

//}

================= ВИКОРИСТАННЯ ШАБЛОННИХ ФУНКЦІЇ =================

//template<typename AnyType, typename Any2>

//AnyType getMax(AnyType a, Any2 b)

//{

// return a \* b ;

//}

//template<typename MType>

//MType getRandomValue()

//{

// return 0 + rand() % 10;

//}

void f()

{

static int r = 10;

printf("r=%d",r);

r++;

}

int main()

{

f();

f();

f();

system("pause");

return 0;

}

// ConsoleApplication23.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

using namespace std;

================ ОБЧИСЛЕННЯ ПОДАТКУ (ЯКЩО НЕ ЗАДАНО, ТО БУДЕ 20%) =========

double getTax(double salary, double taxPercantage = 20)

{

return salary\*taxPercantage / 100;

}

int main()

{

double salary;

printf("Salary=");

cin >> salary;

double tax = getTax(salary); //taxPercantage = 20

printf("Tax=%.2f",tax);

double tax2 = getTax(salary,18); //taxPercantage = 18

printf("Tax2=%.2f", tax2);

system("pause");

return 0;

}

// ConsoleApplication23.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

using namespace std;

================== ПЕРЕДАЄТЬСЯ СПИСОК ДОВІЛЬНОЇ КІЛЬОСТІ ЧИСЕЛ (ВИБРАТИ ДОВІЛЬНЕ З НИХ) ======

int getRandomNumberFromList(int numbersCount, int firstNumber, ...)

{

int\* numbers = &firstNumber;

return numbers[ rand()%numbersCount ]; //rand()%numbersCount - індекс елемента (число від 0 до numbersCount-1)

}

int main()

{

int randomNumber = getRandomNumberFromList(5, 34,21,-19,45,23);

printf("Random from list (34,21,-19,45,23) : %d",randomNumber);

system("pause");

return 0;

}

// ConsoleApplication23.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

using namespace std;

======================= ПЕРЕДАЧА ФУНКЦІЙ ЧЕРЕЗ ПОКАЖЧИКИ ================

==== ЗНАЙТИ АБО СУМУ ДОДАТНИХ, АБО СУМУ ВІД’ЄМНИХ, АБО СУМУ КРАТНИХ 3 ===

//-------------- Описуємо окремі операції (критерії або ж правила)

bool isNumberNegative(int number)

{

return number < 0;

}

bool isNumberPositive(int number)

{

return number > 0;

}

bool isNumberDivBy3(int number)

{

return number % 3 == 0;

}

//-------------- Описуємо загальну функцію

int getSum(int \* a, int n, bool (\*criterion)(int )) //Передаємо масив "а", кількість "n", функцію-критерій "criterion"

{

int s = 0;

for (int i = 0; i < n; i++)

{

if (criterion( a[i] ))

s += a[i];

}

return s;

}

int main()

{

int\* a = new int[5] {12, -9, 10, 18, 4};

printf("1.Sum positive \n");

printf("2.Sum negative \n");

printf("3.Sum div 3 \n");

printf("Your chice: ");

int answer;

cin >> answer;

int result;

switch (answer)

{

case 1: result = getSum(a, 5, isNumberPositive);

break;

case 2: result = getSum(a, 5, isNumberNegative);

break;

case 3: result = getSum(a, 5, isNumberDivBy3);

break;

}

printf("Sum=%d",result);

system("pause");

return 0;

}

// ConsoleApplication23.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

using namespace std;

============== ЗНАЙТИ АБО СУМУ ЗАРОБІТНИХ ПЛАТ, ЩО БІЛЬШІ ЗА МІН.ЗАР.ПЛАТУ, АБО МЕНШІ ===

//-------------- Описуємо окремі операції (критерії або ж правила)

bool isSalaryGT(double salary, double minSalaryValue)

{

return salary>minSalaryValue;

}

bool isSalaryLT(double salary, double minSalaryValue)

{

return salary<minSalaryValue;

}

//-------------- Описуємо загальну функцію

int getSum(double \* a, int n, double minSalaryValue, bool (\*criterion)(double,double)) //Передаємо масив "а", кількість "n", функцію-критерій "criterion"

{

double s = 0;

for (int i = 0; i < n; i++)

{

if (criterion( a[i], minSalaryValue ))

s += a[i];

}

return s;

}

int main()

{

double\* a = new double[5] {10002, 9000, 1000, 1800, 4000};

printf("1.Sum > min \n");

printf("2.Sum < min\n");

printf("Your chice: ");

int answer;

cin >> answer;

int result;

switch (answer)

{

case 1: result = getSum(a, 5, 3200, isSalaryGT);

break;

case 2: result = getSum(a, 5, 3200, isSalaryLT);

break;

}

printf("Sum=%f",result);

system("pause");

return 0;

}

// ConsoleApplication23.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

using namespace std;

==== НАРАХУВАТИ НАДБАВКУ ЗА ОДНИМ З ПРАВИЛ 1 АБО 2 =====================

//-------------- Описуємо окремі операції (критерії або ж правила)

double rule1(double salary)

{

if (salary > 5000)

return 1000;

else

return 2000;

}

double rule2(double salary)

{

if (salary > 2000 && salary<7000)

return 1500;

else

return 0;

}

//-------------- Описуємо загальну функцію

void addExtraMoney(double \* a, int n, double (\*rule)(double ) )

{

for (int i = 0; i < n; i++)

{

a[i] = a[i] + rule(a[i]);

}

}

int main()

{

double\* a = new double[5] {10002, 9000, 1000, 1800, 4000};

printf("1.Add extra (rule 1) \n");

printf("2.Add extra (rule 2) \n");

printf("Your chice: ");

int answer;

cin >> answer;

switch (answer)

{

case 1: addExtraMoney(a, 5, rule1);

break;

case 2: addExtraMoney(a, 5, rule2);

break;

}

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

{

printf("%.8.2f",a[i]);

}

system("pause");

return 0;

}

// ConsoleApplication23.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include<iostream>

using namespace std;

================ СТВОРИТИ ФУНКЦІЮ УПОРЯДКУВАННЯ АБО ЗА ЗРОСТАННЯМ АБО ЗА СПАДАННЯМ ===========

//-------------- Описуємо окремі операції (критерії або ж правила)

bool increaseRule(double number1, double number2 )

{

return number1 > number2;

}

bool decreaseRule(double number1, double number2)

{

return number1 < number2;

}

//-------------- Описуємо загальну функцію

void sort(double \* a, int n, bool (\*compareRule)(double , double ))

{

bool change;

do

{

change = false;

for (int i = 1; i < n; i++)

{

if ( compareRule( a[i-1],a[i]) )

{

swap(a[i-1],a[i]);

change = true;

}

}

} while (change);

}

int main()

{

double\* a = new double[5] {10002, 9000, 1000, 1800, 4000};

printf("1.Increase \n");

printf("2. Decrease \n");

printf("Your choice: ");

int answer;

cin >> answer;

switch (answer)

{

case 1: sort(a, 5, increaseRule);

break;

case 2: sort(a, 5, decreaseRule);

break;

}

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

{

printf("%10.2f -",a[i]);

}

system("pause");

return 0;

}