МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

УНІВЕРСИТЕТ БАНКІВСЬКОЇ СПРАВИ

Лабораторна робота №3

з дисципліни : Об’єктно-орієнтоване програмування

Виконала студентка

1 курсу (скороченого терміну навчання)

Сорока Олена Олександрівна

КИЇВ 2020

**Lab 3.2.1 Vectors and pointers: first try**

**Code:**

#include <iostream>

using namespace std;

int main()

{

int vector[] = { 3, -5, 7, 10, -4, 14, 5, 2, -13 }, \*m = new int, index;

int n = sizeof(vector) / sizeof(vector[0]);

\*m = 1000;

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

if (vector[i] < \*m)

{

\*m = vector[i];

index = i + 1;

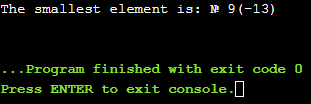
}

cout << "The smallest element is: № " << index << "(" << \*m << ")" << endl;

delete m;

return 0;

}



**Lab 3.2.2 Matrices and pointers – a step inside**

**Code:**

#include <iostream>

using namespace std;

int main()

{

cout<<"Your matrix: "<<endl;

for (int i = 1; i <= 10; i++)

{

for (int j = 1; j <= 10; j++)

{

cout.width(4);

cout << i \* j;

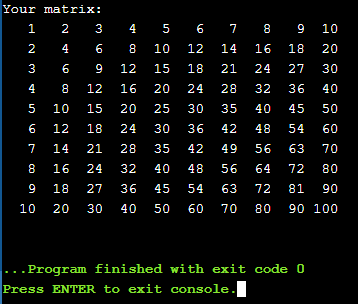
}

cout << endl;

}

return 0;

}

****

**Lab 3.4.1 Old problems, new methods: functions**

**Code:**

**bool isLeap(unsigned int year)**

**{**

**if (year % 400 == 0)**

**return true;**

**if (year % 4 == 0)**

**return true;**

**return false;**

**}**

#include <iostream>

using namespace std;

int main()

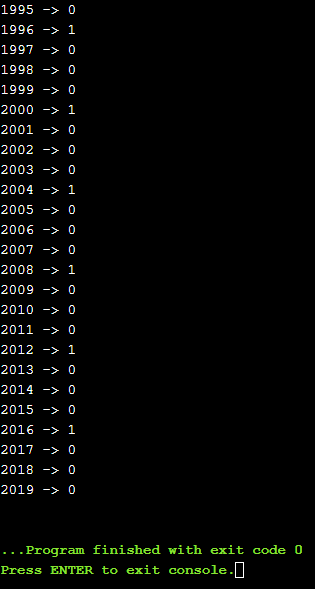
{

for (int yer = 1995; yer < 2020; yer++)

cout << yer << " -> " << isLeap(yer) << endl;

return 0;

}

****

**Lab 3.4.2 One step further: finding the lengths of months**

**Code:**

#include <iostream>

using namespace std;

bool isLeap(int year)

{

if (year % 400 == 0)

return true;

if (year % 4 == 0)

return true;

return false;

}

**unsigned int monthLength\_noCout(int year, int month)**

{

switch (month)

{

case 1:

{

return 31;

break;

}

case 2:

{

if (isLeap(year) == true)

return 29;

else

return 28;

break;

}

case 3:

{

return 31;

break;

}

case 4:

{

return 30;

break;

}

case 5:

{

return 31;

break;

}

case 6:

{

return 30;

break;

}

case 7:

{

return 31;

break;

}

case 8:

{

return 31;

break;

}

case 9:

{

return 30;

break;

}

case 10:

{

return 31;

break;

}

case 11:

{

return 30;

break;

}

case 12:

{

return 31;

break;

}

default:

break;

}

return 0;

}

int main()

{

for (int yr = 2000; yr < 2002; yr++)

{

cout << "---------------------" << endl;

cout << "YEAR : " << yr << endl;

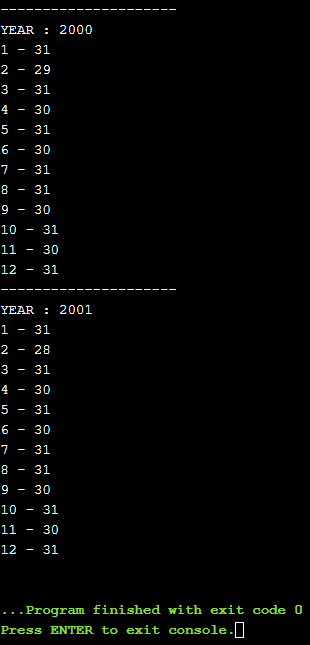
for (unsigned int mo = 1; mo <= 12; mo++)

cout << mo << " - " << monthLength\_noCout(yr, mo) << endl;

}

return 0;

}

****

**Lab 3.4.3 Second step further: finding day of year**

**Code:**

#include <iostream>

using namespace std;

struct Date

{

int year;

int month;

int day;

};

bool isLeap(int year)

{

if (year % 400 == 0)

return true;

if (year % 4 == 0)

return true;

return false;

}

int monthLength(int year, int month)

{

switch (month)

{

case 1:

{

return 31;

break;

}

case 2:

{

if (isLeap(year) == true)

return 29;

else

return 28;

break;

}

case 3:

{

return 31;

break;

}

case 4:

{

return 30;

break;

}

case 5:

{

return 31;

break;

}

case 6:

{

return 30;

break;

}

case 7:

{

return 31;

break;

}

case 8:

{

return 31;

break;

}

case 9:

{

return 30;

break;

}

case 10:

{

return 31;

break;

}

case 11:

{

return 30;

break;

}

case 12:

{

return 31;

break;

}

default:

break;

}

return 0;

}

**int dayOfYear(Date date)**

**{**

**int count = 0;**

**for (int mo = 0; mo <= date.month - 1; mo++)**

**count += monthLength(date.year, mo);**

**count += date.day;**

**return count;**

**}**

int main()

{

Date d;

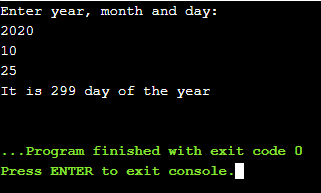
cout << "Enter year, month and day:" << endl;

cin >> d.year >> d.month >> d.day;

cout << "It is " << dayOfYear(d) << " day of the year" << endl;

return 0;

}

****

**Lab 3.4.4 Third step further – counting the days**

**Code:**

#include <iostream>

using namespace std;

struct Date

{

int year;

int month;

int day;

};

bool isLeap(unsigned int year)

{

if (year % 400 == 0)

return true;

if (year % 4 == 0)

return true;

return false;

}

unsigned int monthLength\_noCout(int year, int month)

{

switch (month)

{

case 1:

{

return 31;

break;

}

case 2:

{

if (isLeap(year) == true)

return 29;

else

return 28;

break;

}

case 3:

{

return 31;

break;

}

case 4:

{

return 30;

break;

}

case 5:

{

return 31;

break;

}

case 6:

{

return 30;

break;

}

case 7:

{

return 31;

break;

}

case 8:

{

return 31;

break;

}

case 9:

{

return 30;

break;

}

case 10:

{

return 31;

break;

}

case 11:

{

return 30;

break;

}

case 12:

{

return 31;

break;

}

default:

break;

}

return 0;

}

int dayOfYear(Date date)

{

int count = 0;

for (int mo = 0; mo <= date.month - 1; mo++)

count += monthLength\_noCout(date.year, mo);

count += date.day;

return count;

}

**int theDiff(Date s, Date t)**

**{**

**if (dayOfYear(t) - dayOfYear(s) >= 0)**

**{**

**if (t.year > s.year)**

**{**

**int g;**

**g = (t.year - s.year) \* 365;**

**return dayOfYear(t) - dayOfYear(s) + g;**

**}**

**if (t.year == s.year)**

**return dayOfYear(t) - dayOfYear(s);**

**}**

**else**

**return -1;**

**}**

int main()

{

Date since;

Date till;

cout << "Enter the 'since' date:" << endl;

cin >> since.year >> since.month >> since.day;

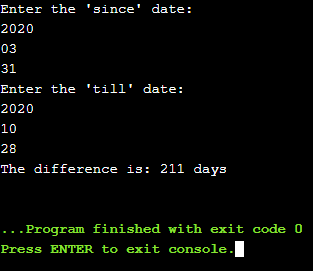
cout << "Enter the 'till' date:" << endl;

cin >> till.year >> till.month >> till.day;

cout << "The difference is: " << theDiff(since, till) << " days" << endl;

return 0;

}

****

**Lab 3.4.5 A foretaste of system programming – obtaining the current date**

**Code:**

#include <iostream>

using namespace std;

struct Date

{

int year;

int month;

int day;

};

bool isLeap(unsigned int year)

{

if (year % 400 == 0)

return true;

if (year % 4 == 0)

return true;

return false;

}

unsigned int monthLength\_noCout(int year, int month)

{

switch (month)

{

case 1:

{

return 31;

break;

}

case 2:

{

if (isLeap(year) == true)

return 29;

else

return 28;

break;

}

case 3:

{

return 31;

break;

}

case 4:

{

return 30;

break;

}

case 5:

{

return 31;

break;

}

case 6:

{

return 30;

break;

}

case 7:

{

return 31;

break;

}

case 8:

{

return 31;

break;

}

case 9:

{

return 30;

break;

}

case 10:

{

return 31;

break;

}

case 11:

{

return 30;

break;

}

case 12:

{

return 31;

break;

}

default:

break;

}

return 0;

}

int dayOfYear(Date date)

{

int count = 0;

for (int mo = 0; mo <= date.month - 1; mo++)

count += monthLength\_noCout(date.year, mo);

count += date.day;

return count;

}

**Date today()**

**{**

**time\_t t = time(NULL);**

**tm tl = \*localtime(&t);**

**Date today;**

**today.year = tl.tm\_year + 1900;**

**today.month = tl.tm\_mon + 1;**

**today.day = tl.tm\_mday+1;**

**return today;**

**}**

int main()

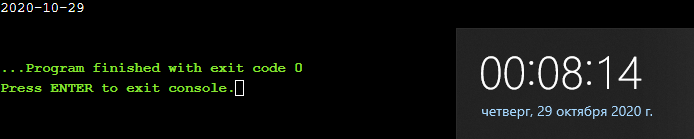
{

Date t = today();

cout << t.year << "-" << t.month << "-" << t.day << endl;

return 0;

}

****

**Lab 3.4.6 Prime numbers – how do we find them?**

**Code:**

#include <iostream>

using namespace std;

bool isPrime(int num)

{

for (int i = 2; i <= num / 2; i++)

if ((num % i) == 0)

return false;

return true;

}

int main()

{

for (int i = 1; i <= 21; i++)

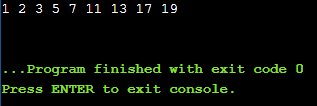
if (isPrime(i))

cout << i << " ";

cout << endl;

return 0;

}

****

**Lab 3.6.1 Modifying a function argument's value – how do we do it?**

**Code:**

#include <iostream>

using namespace std;

void increment(int\* var)

{

\*var++;

}

void increment(int\* var, int v)

{

\*var += v;

}

void increment(float& floatVar)

{

floatVar++;

}

void increment(float& floatVar, int intVar)

{

floatVar += intVar;

}

int main()

{

int var = 0;

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

if (i % 2 == 0)

increment(&var);

else

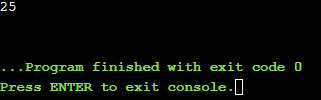
increment(&var, i);

}

cout << var << endl;

return 0;

}



**Lab 3.8.1 Overloading functions**

**Code:**

#include <iostream>

#include<math.h>

using namespace std;

void increment(int\* var)

{

\*var++;

}

void increment(int\* var, int v)

{

\*var += v;

}

void increment(float& floatVar)

{

floatVar++;

}

void increment(float& floatVar, int intVar)

{

floatVar += intVar;

}

int main()

{

int intvar = 0;

float floatvar = 1.5;

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

if (i % 2 == 0)

{

increment(&intvar);

increment(floatvar, sqrt(intvar));

}

else

{

increment(&intvar, i);

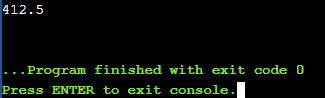
increment(floatvar);

}

cout << intvar \* floatvar << endl;

return 0;

}



**Lab 3.10.1 Overloading functions**

**Code:**

#include <iostream>

#include<cstdlib>

#include<ctime>

using namespace std;

int main()

{

unsigned int maxball, ballsno, num;

cout << "Max ball number? ";

cin >> maxball;

cout << "How many balls? ";

cin >> ballsno;

unsigned int\* randBall = new unsigned int[ballsno];

srand(time(NULL));

for (unsigned int i = 0; i < ballsno; i++)

{

num = rand() % maxball + 1;

randBall[i] = num;

cout << randBall[i] << " ";

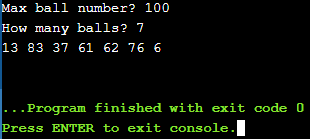
}

cout << endl;

delete[] randBall;

return 0;

}

****

**Lab 3.10.2 Dynamic data – how to obtain it and how to get rid of it**

**Code:**

#include <iostream>

#include<cstdlib>

#include<ctime>

using namespace std;

struct Collection

{

unsigned int elno;

unsigned int\* elements;

};

void AddToCollection(Collection& col, unsigned int element, int i)

{

if (col.elno == 0)

col.elements[0] = 0;

else

col.elements[i] = element;

}

void PrintCollection(Collection col)

{

cout << "[ ";

for (unsigned int i = 0; i < col.elno; i++)

cout << col.elements[i] << " ";

cout << "]" << endl;

}

int main()

{

Collection collection = { 0, NULL };

unsigned int elems;

cout << "How many elements? ";

cin >> elems;

srand(time(NULL));

collection.elno = elems;

collection.elements = new unsigned int[collection.elno];

for (unsigned int i = 0; i < elems; i++)

AddToCollection(collection, rand() % 100 + 1, i);

PrintCollection(collection);

delete[] collection.elements;

return 0;

}

