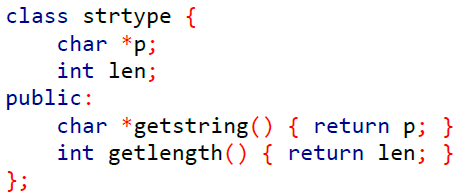
**ПРАКТИЧНЕ ЗАВДАННЯ № 6**

**Завдання №1 :**

Дано неполное определение класса:



Добавьте в это определение два конструктора. В первом не должно быть параметров. Он должен выделять 255 байтов памяти (с помощью оператора new), инициализировать эту память нулевой строкой и устанавливать переменную len равной 255. Во втором конструкторе должно быть два параметра. Первый — это строка, используемая при инициализации, второй — число выделяемых байтов. Во второй версии конструктора должно выделяться заданное количество памяти, в которую должна помещаться копия строки. Необходимо реализовать полный контроль границ массива (подсказка: см. Unit 5) и, разработать в короткую программу вывода, показать, что оба конструктора работают так, как это было задумано.

**Код програми:**

#include <iostream>

#include <string.h>

using namespace std;

class String {

private:

char \*p;

int len;

public:

char \*getstring () { return p; }

int getlenght () { return len; }

String(){

len = 255;

p = new char[255];

if(!p)

exit(1);

\*p = '\0';

}

String(char const \*str, int size){

if(strlen(str) >= size)

exit(1);

len = size;

p = new char[size];

if(!p)

exit(1);

strcpy(p, str);

}

~String() {

delete p;

}

// show() {

// cout << p << ' ' << len << endl;

// }

};

int main() {

int a;

char str[100];

String s1;

cin >> str >> a;

String s2(str, a);

//String s2("HELLO!", a);

cout << s1.getstring() << ' ' << s1.getlenght() << endl;

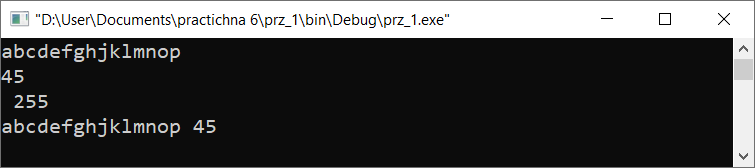
cout << s2.getstring() << ' ' << s2.getlenght() << endl;

// s2.show();

return 0;

}

**Результати роботи програми (скриншоти)**



**Завдання №2 :**

В Unit 2 (см. файл Pract02.pdf, Задача 2.14) вы создали имитацию секундомера. Модифицируйте ваше решение так, чтобы в классе stopwatch был и конструктор без параметров (как это уже сделано) и его перегруженная версия для доступа к системному времени через стандартную функцию clock(). Покажите, что внесенные изменения работают. Совет от ВВ: Подумайте о том, каким образом перегруженный конструктор может быть полезен для ваших собственных программных задач, например, в курсовой работе.

**Код програми:**

#include <iostream>

#include <ctime>

#include <windows.h>

using namespace std;

class Watch {

clock\_t start\_p;

clock\_t start\_time;

public:

Watch();

***Watch(clock\_t time){***

***start\_p = time;***

***}***

~Watch();

void start();

void show();

};

Watch::Watch() {

start\_p = clock();

}

void Watch::start() {

start\_time = clock();

}

void Watch::show() {

cout << (double)(clock() - start\_time) / CLOCKS\_PER\_SEC << endl;

}

Watch::~Watch() {

cout << (double)(clock() - start\_p) / CLOCKS\_PER\_SEC << endl;

}

int main() {

srand(time(NULL));

Watch timer;

int num;

cin >> num;

timer.start();

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

cout << rand() % 64 << ' ';

if (i == 4) {

Sleep(num \* 1000);

puts("\n");

}

}

puts("\n");

timer.show();

puts("\n");

Watch timer2(clock());

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

cout << rand() % 16 << ' ';

if (i == 4) {

Sleep(num \* 1000);

puts("\n");

}

}

puts("\n");

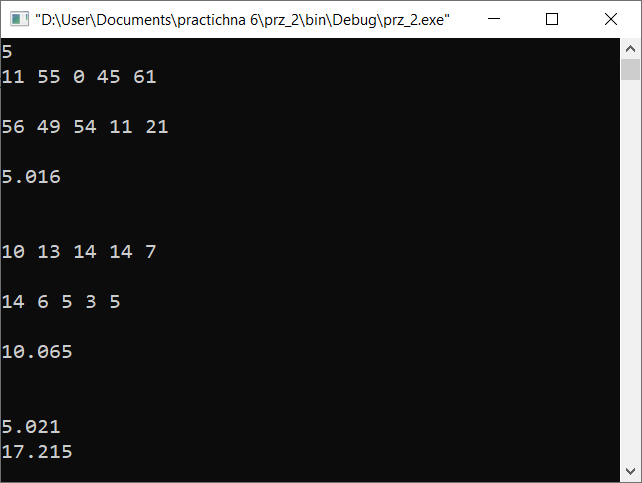
timer.show();

puts("\n");

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання №3 :**

***Конструктор копий вызывается и в тех случаях,*** когда функция генерирует временный объект, используемый в качестве ее возвращаемого значения (для тех функций, которые возвращают объекты). Зная это, рассмотрим следующий результат работы программы:

Constructing normally

Constructing normally

Constructing copy

Constructing copy

Эти строки появились в результате работы следующей программы:

**Код програми:**

#include <iostream>

using namespace std;

class myclass {

public:

myclass();

myclass(const myclass &obj);

myclass f();

myclass g(myclass obj);

};

// Normal constructor

myclass::myclass() {

cout << "Constructing normally\n";

}

// Copy constructor

myclass::myclass(const myclass &obj) {

cout << "Constructing copy\n";

}

// Return an object.

myclass myclass::f() {

myclass temp;

return temp;

}

myclass myclass::g(myclass obj) {

myclass temp = obj;

return temp;

}

int main() {

myclass obj;

obj = obj.f();

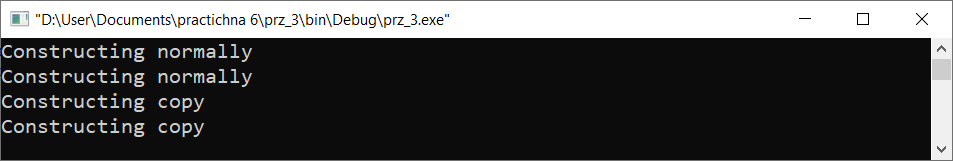
obj = obj.g(obj);

return 0;

}

/\*Об’єкти ***obj temp*** створюються за допомогою звичайного конструктора, але у функція ***obj = obj.f();*** повертає об’єкт ***temp,*** тому генерується тимчасовий об’єкт, який викликає конструктор копіювання.\*/

**Результати роботи програми (скриншоти):**



**Завдання №4 :**

***Объясните, почему следующая программа заканчивается сообщением***

***Aborted (core dumped) и исправьте ее:***

**Код програми:**

#include <iostream>

#include <cstdlib>

using namespace std;

class myclass {

private:

int \*ptr;

public:

myclass(int i);

~myclass() { delete ptr; }

friend int getval(myclass &obj);

};

myclass::myclass(int i){

ptr = new int(i);

if (!ptr)

exit(1);

}

int getval(myclass ***&obj***) {

return \*obj.ptr;

}

int main() {

myclass a(1), b(2);

cout << getval(a) << ' ' << getval(b) << endl;

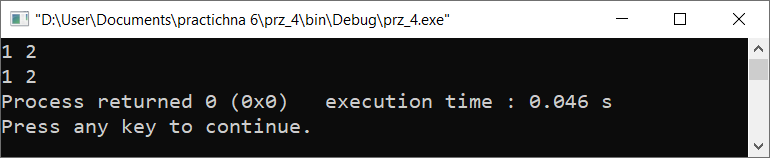
cout << getval(a) << ' ' << getval(b);

return 0;

}

/\*При передачі об'єкта у функцію getval() створюється його копія, але Функція getval() повертає значення копія видаляється, а пам'ять виділена для об'єкта який передається getval(myclass obj) звільнюється і в подальшому вона не може бути використана, тому виникає помилка\*/

**Результати роботи програми (скриншоти):**



**Завдання №5 :**

В стандартной библиотеке C++ существует функция strtol(), имеющая следующий прототип: long strtoi(const char \*start, const \*\*end, int base);

Функция преобразует обозначающую число строку, на которую ссылается указатель start, в длинное целое. Число base задает основание системы счисления этого числа. При возвращении функцией своего значения указатель end ссылается на символ в строке, следующий сразу за последней цифрой строки. озвращаемое длинное целое эквивалентно тому числу, которое записано в строке. Диапазон значений base от 2 до 38. Однако наиболее часто основание системы счисления равно 10. Создайте функцию mystrtol(), работающую точно так же, как и функция strtol(), но аргумент 10 должен передаваться параметру base по умолчанию. азрешается свободно пользоваться функцией strtol() для фактического преобразования. Для этого в программу требуется включить заголовок <cstdlib>. Покажите, что ваша версия работает правильно.

**Код програми:**

#include <iostream>

#include <cstdlib>

using namespace std;

long mystrtol(const char \*start, char \*\*end){

return strtol(start, end, 10);

}

long mystrtol(const char \*start, char \*\*end, int base){

return strtol(start, end, base);

}

int main() {

char \*p;

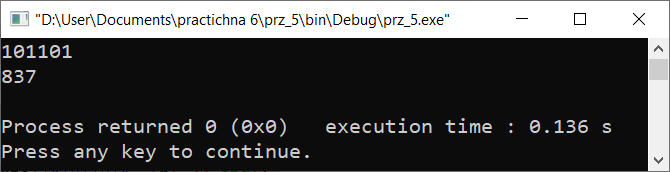
cout << mystrtol("101101", &p) << endl;

cout << mystrtol("345", &p, 16) << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання №6 :**

В большинстве компиляторов C++ применяются нестандартные функции, управляющие позиционированием курсора и другими аналогичными действиями. mВы помните, что вам ранее уже было разрешено пользоваться любыми компиляторами, а не только g++? Если в используемом вами компиляторе применяются такие функции, создайте функцию myclreol(), которая стирает строку, начиная от текущей позиции курсора до конца строки. Передайте этой функции параметр, задающий число стираемых позиций. Если параметр не задавать, то по умолчанию должна стираться вся строка. В противном случае должно стираться число символьных позиций, заданное параметром.

**Код програми:**

#include <iostream>

#include <cstring>

#include <windows.h>

using namespace std;

void myclreol(int cursor, int len){

COORD pos;

//pos.Y = 5;

pos.Y = 3;

pos.X = cursor;

for (int i = SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), pos); i < len - pos.X + 1; ++i)

cout << ' ';

}

int main(){

string str;

int a;

// cout << "enter string" << endl;

getline(cin, str);

cout << str << endl;

// cout << "enter position" << endl;

cin >> a;

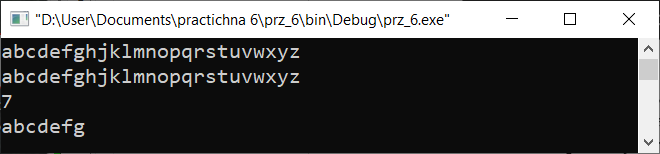
cout << str << endl;

myclreol(a, str.length());

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання №7 :**

***Попытайтесь провести компиляцию всех программ (см. Unit6, "Overloading And Ambiguity"), в которых имеет место неоднозначность.*** Сопоставьте сообщения компилятора, объясните и запомните эти сообщения об ошибках. Это поможет вам сразу распознать ошибки неоднозначности, если они появятся в ваших программах.

**a)**

**Код програми:**

#include <iostream>

using namespace std;

float f(float t) {

return t /2.0;

}

double f(double t){

return t/3.0;

}

int main(){

float x = 10.09;

cout<<f(x);

double y = 10.09;

cout<<f(y);

cout<<f(10);

return 0;

}

**/\*Неоднозначність при перезагрузці функцій**

При виклику функції ***f(10);*** передається аргумент типу int має відбуватись приведення типу, але через те, що типи float і double аналогічні виникає неооднозначність – компілятор не знає яку функцію викликати\*/

**Результати роботи програми (скриншоти):**



**b)**

#include <iostream>

using namespace std;

void f(unsigned char c) {

cout << c;

}

void f(char c){

cout << c;

}

int main() {

f('c');

f(86);

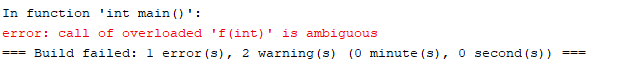
return 0;

}

/\*Неоднознаність при виклику функції з неправильним типом ***f(86);***

Компілятор не знає, яку функцію викликати для приведення типу, адже f(unsigned char) і f(char) аналогічні***\*/***

**Результати роботи програми (скриншоти):**

****

**c)**

**Код програми:**

#include <iostream>

using namespace std;

int f(int a, int b) { return a + b; }

int f(int a, int &b) { return a - b; }

int main() {

int x = 1, y = 2;

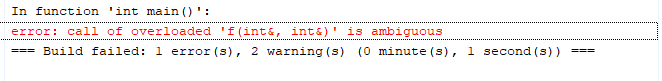
cout << f(x, y);

return 0;

}

/\*Неоднозначність при перезагрузці функцій, одна з яких використовує посилання, інша –значення. Синтаксично ці функції не відрізняються, тому виникає неоднозначність і компілятор не знає, яку з ункцій викликати\*/

**Результати роботи програми (скриншоти):**

****

**d)**

**Код програми:**

#include <iostream>

using namespace std;

int f(int a) { return a \* a; }

int f(int a, int b = 0) { return a \* b; }

int main() {

cout << f(10, 2);

cout << f(10);

return 0;

}

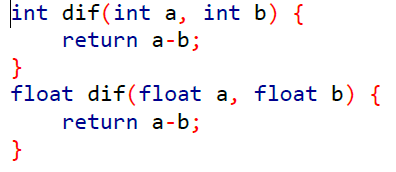
**/\***Привиклику функції f(10) компілятор не знає,яку версію функції вибрати f(int a) чи f(int a, int b = 0)  **\*/**

**Результати роботи програми (скриншоти):**

****

**Завдання №8 :**

Ниже приведены две перегруженные функции. Покажите в демонстрационной программе, как получить и как использовать адрес каждой из них:



**Код програми:**

#include <iostream>

using namespace std;

int f(int a, int b) {

return a - b;

}

float f(float a, float b) {

return a - b;

}

int main() {

int (\* p1)(int, int);//creating a pointer to function

float (\* p2)(float, float);

p1 = f;

cout << p1(87, 4) << endl;

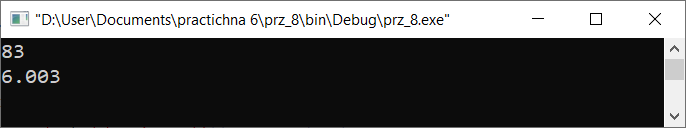
p2 = f;

cout << p2(7.557, 1.554) << endl;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання №9 :**

***Перегрузите конструктор Date()*** ***(См. Unit6 "Select the most convenient method of initializing an object") так, чтобы он имел параметр типа time\_t. Подсказка: Вспомните, что time\_t - это тип данных, определенный стандартными библиотечными функциями времени и даты компилятора C++.***

**Код програми:**

#include <iostream>

#include <cstdio>

#include <ctime>

using namespace std;

class Date {

private:

int day, month, year;

public:

Date(int m, int d, int y) {

month = m;

day = d;

year = y;

}

Date(const char \*str){

sscanf(str, "%d%\*c%d%\*c%d", &month, &day, & year);

}

Date(time\_t info){

struct tm \*p;

time(&info);

p = localtime(&info);

//p = gmtime(&info);

month = p -> tm\_mon;

day = p -> tm\_mday;

year = p -> tm\_year;

}

void show(){

cout << month << '/' << day << '/' << year << endl;

}

};

int main(){

Date t1("19/03/21");

t1.show();

Date t2(19, 03, 21);

t2.show();

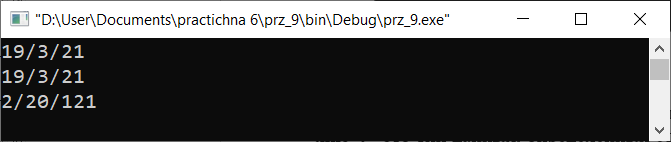
Date t3(time(NULL));

t3.show();

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання №10 :**

Создайте функцию reverse() с двумя параметрами. Первый параметр str - это указатель на строку, порядок следования символов в которой, после возвращения функцией своего значения, должен быть заменен на обратный. Второй параметр count задает количество переставляемых в строке str символов. Значение count по умолчанию должно быть таким, чтобы в случае его не задания функция reverse() меняла порядок следования символов в целой строке.

**Код програми:**

#include <iostream>

#include <cstring>

using namespace std;

void reverse\_s(char str[], int n){

for(int j = 0, k = n - 1; j < k; j++, k--)

swap(str[j],str[k]);

}

void reverse\_s(char str[]) {

for(int j = 0, k = strlen(str) - 1; j < k; j++, k--)

swap(str[j],str[k]);

}

int main (){

char \*str = new char[100];

char \*str1 = new char[100];

int n;

cin.getline(str, 100);

reverse\_s(str);

cout << str << endl;

cin.getline(str1, 100);

cin >> n;

reverse\_s(str1, n);

cout << str1 << endl;

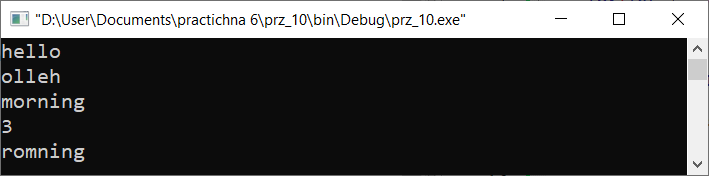
delete [] str;

delete [] str1;

return 0;

}

**Результати роботи програми (скриншоти):**



**Завдання №11 :**

Создайте функцию order(), которая получает два параметра-ссылки на целые. Если первый аргумент больше второго, поменяйте их значения. В противном случае ничего делать не надо. Таким образом, порядок следования двух аргументов, используемых при вызове функции order(), должен быть таким, чтобы всегда после возвращения функцией своего значения первый аргумент был меньше второго. Например, если дано



то после вызова функции х будет равен 0, а у будет равен 1.

**Код програми:**

#include <iostream>

using namespace std;

void order(int &a, int &b) {

if (a > b)

swap(a, b);

}

int main() {

int a, b;

cin >> a >> b;

//cout << a << ' ' << b << endl;

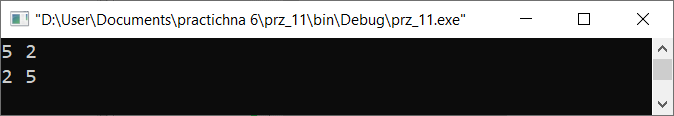
order (a, b);

cout << a << ' ' << b << endl;

return 0;

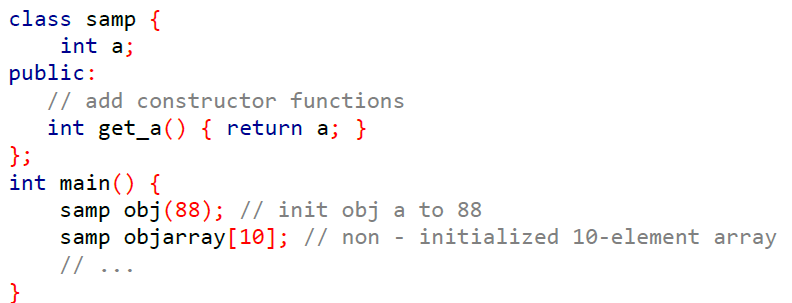
}

**Результати роботи програми (скриншоти):**



**Завдання №12 :**

Пусть дано следующее неполное описание класса, добавьте конструкторы так, чтобы оба объявления в функции main() были правильны. Подсказка: вам необходимо дважды перегрузить конструктор samp().



**Код програми:**

#include <iostream>

using namespace std;

class Object {

private:

int a;

public:

Object() { a = 0; }

Object(int n) { a = n; }

int get\_a() { return a; }

};

int main() {

Object obj(88); // init obj a to 88

Object objarray[10]; // non - initialized 10-element array

cout << obj.get\_a() << endl;

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

// cout << objarray[i].get\_a() << ' ';

return 0;

}

**Результати роботи програми (скриншоти):**

