When must you use a member initializer list

What does a compiler add to an empty class declaration in C++ ?

Which of these functions is a valid const member function signature?

What is the effect of the unnamed namespace?

What is an abstract base class?

What is an rvalue?

Why might you declare member data mutable?

In which order are the member data of a class initialised in a constructor?

What lines have boxing?

What is a pure virtual function?

When should a class have a virtual destructor?

Why is a virtual function call slower to execute than a statically bound call?

#define min(x,y) (x<y) ? x : y

How much memory will be allocated for 1000 bool variables stored in a

std::vector<> and std::list<>

What is the output of the following code?

int num = 0;

bool out = num == 0;

std::cout << out << std::endl;

What is the value of 'c'?

int a = 5;

int b = 2;

auto c = a / b;

What is the value of 'n'?

const char message[] =

{ '1', '2', '3', '4', '5', '\0' };

auto n = strlen(message);

If no other defines are passed at compilation what would be the output after calling func()?

#define FOO 1

#define BAR 0

void func()

{

#if FOO && BAR

printf("W");

#elif FOO == 0 && BAR == 1

printf("X");

#elif defined(FOO) && defined(BAR)

printf("Y");

#else

printf("Z");

#endif

}

What would be printed by the following code?

int foo = 42;

printf("%d\n", foo++);

What will the value of 'foo' be at the end of the following code:

int foo = 0;

++foo -= 2;

if(foo>=0) foo+=1;

What would sizeof(Foo) return?

struct Foo

{

Foo();

int get(){ return 42; }

};

What is sizeof(Foo)?

struct Foo

{

Foo();

char get(int i){ return x[i]; }

char x[5];

};

What will be the value of 'a'?

int a = 3 + 4 \* 2 / 5;

What is the output of the following?

int Square(int x)

{

return x \* x;

}

#define SQUARE(x) (x \* x)

int main()

{

std::cout << Square(2.0f + 3.5f) << ", " << SQUARE(2.0f + 3.5f) << std::endl;

}

<https://awesomeopensource.com/project/SergeyMakeev/ArcadeCarPhysics>