**Quizzzzzzzzzzzzzzzzz!**

----------------------------------------------------------------

1.

#include<iostream>

using namespace std;

class Base1 {

public:

Base1()

{ cout << " Base1's constructor called" << endl; }

};

class Base2 {

public:

Base2()

{ cout << "Base2's constructor called" << endl; }

};

class Derived: public Base1, public Base2 {

public:

Derived()

{ cout << "Derived's constructor called" << endl; }

};

int main()

{

Derived d;

return 0;

}

A.

Compiler Dependent

B.

Base1′s constructor called

Base2′s constructor called

Derived’s constructor called

C.

Base2′s constructor called

Base1′s constructor called

Derived’s constructor called

D.

Compiler Error

----------------------------------------------------------------

2.

Assume that an integer takes 4 bytes and there is no alignment in following classes, predict the output.

#include<iostream>

using namespace std;

class base {

int arr[10];

};

class b1: public base { };

class b2: public base { };

class derived: public b1, public b2 {};

int main(void)

{

cout << sizeof(derived);

return 0;

}

A.40

B.80

C.0

D.4

----------------------------------------------------------------

3.

#include<iostream>

using namespace std;

class P {

public:

void print() { cout <<" Inside P"; }

};

class Q : public P {

public:

void print() { cout <<" Inside Q"; }

};

class R: public Q { };

int main(void)

{

R r;

r.print();

return 0;

}

A.Inside P

B.Inside Q

C.Compiler Error: Ambiguous call to print()

----------------------------------------------------------------

4.

#include<iostream>

using namespace std;

class Base {

private:

int i, j;

public:

Base(int \_i = 0, int \_j = 0): i(\_i), j(\_j) { }

};

class Derived: public Base {

public:

void show(){

cout<<" i = "<<i<<" j = "<<j;

}

};

int main(void) {

Derived d;

d.show();

return 0;

}

A.i = 0 j = 0

B.Compiler Error: i and j are private in Base

C.Compiler Error: Could not call constructor of Base

----------------------------------------------------------------

5.

#include<iostream>

using namespace std;

class Base

{

public:

int fun() { cout << "Base::fun() called"; }

int fun(int i) { cout << "Base::fun(int i) called"; }

};

class Derived: public Base

{

public:

int fun() { cout << "Derived::fun() called"; }

};

int main()

{

Derived d;

d.fun(5);

return 0;

}

A.Base::fun(int i) called

B.Derived::fun() called

C.Base::fun() called

D.Compiler Error

----------------------------------------------------------------

6.

#include<iostream>

using namespace std;

class Base {

public:

int fun() { cout << "Base::fun() called"; }

int fun(int i) { cout << "Base::fun(int i) called"; }

};

class Derived: public Base {

public:

int fun() { cout << "Derived::fun() called"; }

};

int main() {

Derived d;

d.Base::fun(5);

return 0;

}

A.Compiler Error

B.Base::fun(int i) called

----------------------------------------------------------------

7.

#include<iostream>

using namespace std;

class Base

{

public :

int x, y;

public:

Base(int i, int j){ x = i; y = j; }

};

class Derived : public Base

{

public:

Derived(int i, int j):x(i), y(j) {}

void print() {cout << x <<" "<< y; }

};

int main(void)

{

Derived q(10, 10);

q.print();

return 0;

}

A.10 10

B.Compiler Error

C.0 0

----------------------------------------------------------------

8.

#include<iostream>

using namespace std;

class Base

{

protected:

int a;

public:

Base() {a = 0;}

};

class Derived1: public Base

{

public:

int c;

};

class Derived2: public Base

{

public:

int c;

};

class DerivedDerived: public Derived1, public Derived2

{

public:

void show() {cout << a;}

};

int main(void)

{

DerivedDerived d;

d.show();

return 0;

}

A.Compiler Error in Line "cout << a;"

B.0

C.Compiler Error in Line "class DerivedDerived: public Derived1, public Derived2"

----------------------------------------------------------------

9.

#include<iostream>

using namespace std;

class Base1

{

public:

char c;

};

class Base2

{

public:

int c;

};

class Derived: public Base1, public Base2

{

public:

void show() { cout << c; }

};

int main(void)

{

Derived d;

d.show();

return 0;

}

A.Compiler Error in "cout << c;"

B.Garbage Value

C.Compiler Error in "class Derived: public Base1, public Base2"

================================================================

**Assignment**

Xây dụng base class, derived class cho shape, triangle, right triangle. Khởi tạo derived triangle instance, derived triangle instance và tính chu vi, diện tích của mỗi instance.

