**PDF 2010-1**

1. Class A{

int n;

private:

A(int k){n=k;}

&A operator=(int op2){

n=op2;

}

&A operator\*=(int op2){

n=n\*op2;

}

&A operator=(&A op2){

n=n\*op2.n;

}

int get(){

if (n==100) n=50;

return n;

}

};

2.

int x = 0;

void f(int a, int b) { x = a+b; }

class A {

int x;

public:

void f() { x = 2; }

void f(char a1, char b1) { x = a1-b1; }

};

class B: public A {

public:

void f(int a) { ::x = a; }

void g () {

f();

f(0);

f(5.3 , 1);

x = 1;

}

};

int main () {

B b;

f(2);

f(3, 'a');

return 0;

}

3.

struct A {

int i;

virtual void f() = 0;

virtual ~A() {}

};

int g(A a) { return a.i \* 5; } а ещё нет конструктора!

4. Шаблоны класса

template <class Cs> void func(const Cs& ref) { /\*...\*/ }

template <int n> void func(int t = n) { /\*...\*/ }

template <double f> void func(double d = f) { /\*...\*/ }э

5.

first

third

result = (118,2)

second

fourth

6.

class A{

public:

static int x;

void get\_0() {

return;

}

}

7. ???

class Cls {

const int i;

public:

Cls() { i = 1; }

};

void f(Cls \*p, Cls \*q) {

\*p = \*q;

}

8.

sun

venus 20

venus 400

moon

earth 420

venus 1000

moon

moon

1429 1420

9.

**Инкапсуляция** - механизм, связывающий вместе код и данные, которыми он манипулирует, и одновременно защищающий их от произвольного доступа со стороны другого кода, внешнего по отношению к рассматриваемому. Доступ к коду и данным жестко контролируется интерфейсом.

**АТД** – класс/структура, в которой есть хотя бы одна чисто виртуальная функция.

**Virtual name(..)=0;**

10.

Cat

Wolf

Cow

**PDF 2010-2**

1.

Class B{

Int I;

Public:

B(): i(10) {}

Void Operator= (const B& op){

I=op.i+10;

}

B(const B& op){

I=op.i+10;

}

};

2.???

double a = 0;

void f(double x = 2) { a = x; }

void f() { a = 1; }

struct B {

double a;

void f(){ a = 2; }

};

class D: B {

public:

void f(int a) { ::a = a; }

void h() {

f('r');

f();

a = 2;

}

};

int main () {

D d;

f();

f(6);

return 0;

}

3.

struct S {

virtual void f() const = 0;

virtual ~S() {}

};

struct A {

S s; - нельзя создать абстрактный класс

int i;

};

4. ???

template <int n> class A { /\*...\*/ };

template <float f> class B { /\*...\*/ };

template <class Cs> class C { /\*...\*/ };

5.

Mercury

Earth

(37-96-3=-62 ; 8+3=11)

Venus

mars

6.

Class C{

Public:

Const Static char c;

Void f(){

Cout << c;

}

};

7.

8.

Owl

Sheep

Sheep

Wolf

Horse

Sheep

Wolf

Wolf

Wolf

9.

Public является частью public’a производного класса, т.е. его члены доступны из любой точки программы, а private-члены недоступны даже внутри производного класса.

10.

Lance

Knife

Hammer

**PDF 2010-3**

1. Class C{

Int k;

Public:

C(int op){k=op;}

Const C& operator=(int op){

C t(2\*op);

Return t;

}

C operator+(const C& op){

C t(2\*(op.k+k));

Return t;

}

}

2.

float y = 0;

void f(float a) { y = a; }

class T {

int y;

public:

void f() { y = 2; }

};

class S : public T {

public:

void f(float n, float m) { ::y = n \* m; }

void f(char c1, char c2) { ::y = c1 + c2; }

void g () {

f();

::f(1);

S::f(-1 , 1);

y = 2;

}

};

int main () {

S b;

::f(5);

f('+', 6);

return 0;

}

3.

class B {

public:

virtual int f() = 0;

int g() { return f() \* 10; }

virtual ~B() {}

};

int h(B b) { return b.g() + 2; }

.4..???

struct mystruct { int a, b; };

template <int n> void func(int t = n) { /\*...\*/ }

template <mystruct a> void func(mystruct \*p = &a) { /\*...\*/ }

template <class Cs> void func(const Cs& ref) { /\*...\*/ }

5.

Dog

Sheep

Result = (33 ; 4)

Cat

Horse

6.

Class X{

Public:

Static void g(){

Cout << “static g” << endl;

}

Void h(){

Cout << “h” << endl;

}

};

7.???

Class smartstr{

Smartstr\* operator new (smartstr[]){

..}

8.

Fist

Lance 30

Lance 700

pistole

Dagger 730

Lance 1000

Pistol

Pistol

1735 730

Pistol

Pistol

10.

Gamma

Epsilon