9. ООП. Дать определение инкапсуляции. Что такое АТД? Каким образом АТД реализуется в C++?

10. Что напечатает следующая программа?

```
class Ex {
    int code;
public:
    Ex(int i) : code(i) {}
    Ex(const Ex& ex) : code(ex.code) {}
    int Get() const { return code; }
};
struct Ex90 : Ex {
    Ex90() : Ex(90) {}
};
void f() {
    throw Ex90();
    printf("dog\n");
}
void t() {
    try { f(); }
    catch(Ex90 &x) {
        printf("cat\n");
        throw Ex(x.Get() + 1);
        printf("sheep\n");
    }
    catch(Ex &) { printf("horse\n"); }
    printf("cow\n");
}
int main() {
    try { t(); }
    catch(Ex &x) { printf("elephant %d\n", x.Get()); }
    catch(...) { printf("wolf\n"); }
    return 0;
}
```

B.1_2010
Ф.И.О._____ гр.____

1	2	3	4	5	6	7	8	9	10

1. Описать класс `A` таким образом, чтобы все конструкции функции `main()` были верными, а на экран выдалось `50 300`.

```
int main () {
    A a1(5), a2 = 3;
    a1 *= 10;
    a2 *= a1 *= 2;
    cout << a1.get() << ' ' << a2.get() << endl;
    return 0;
}
```

Изменять функцию `main`, использовать механизм исключений, а также функции `exit()`, `terminate()` и т. п. для досрочного завершения программы **запрещается**.

2. Если в реализации функций `B::g()` и `main()` есть ошибки, **объясните**, в чем они заключаются. В теле функций `main()` и `B::g()` к именам каждой правильной конструкции добавьте с помощью символа `::` префикс, указывающий, с какой областью видимости связано имя.

```
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. Дан текст фрагмента программы:

```
struct A {
   int i;
   virtual void f() = 0;
   virtual ~A() {}
};
int g(A a) { return a.i * 5; }
```

Есть ли в этом тексте ошибки? Если да, то в чём они заключаются?

4. Есть ли ошибки в следующих заголовках шаблонов? Если есть, поясните, в чём они заключаются.

```
template <class Cs> void func(const Cs& ref) { /*...*/ }
template <int n> void func(int t = n) { /*...*/ }
template <double f> void func(double d = f) { /*...*/ }
```

5. Что напечатает следующая программа?

```
class A {
    int i;
public:
    A(int x) { i = x; printf("first\n"); }
    virtual ~A() { printf("second\n"); }
    int f() const { return i + g() + h(); }
    virtual int g() const { return i; }
    int h() const { return 39; }
};
class B : public A {
public:
    B() : A(70) { printf("third\n"); }
    ~B() { printf("fourth\n"); }
    int f() const { return g() - 2; }
    virtual int g() const { return 4; }
    int h() const { return 6; }
};
int main() {
    B b;
    A* p = &b;
    printf("result = (%d ; %d)\n", p->f(), b.f());
    return 0;
}
```

6. Описать класс A таким образом, чтобы были верными все конструкции следующего фрагмента программы:

```
int A::x;
int main () {
    const A a;
    a.x = 1;
    a.get_0();
    return 0;
}
```

7. Даны описание класса и функции:

```
class Cls {
    int i;
public:
    Cls() { i = 1; }
};
void f(Cls *p, Cls *q) {
    *p = *q;
}
```

Дополните описание класса `Cls` (не изменяя описание функции `f`) так, чтобы только описание `f` стало ошибочным.

8. Что напечатает следующая программа?

```
class I {
    int i;
public:
    I() : i(9) { printf("sun\n"); }
    I(int a) : i(a) { printf("venus %d\n", i); }
    I(const I& other) : i(other.i) { printf("earth %d\n", i); }
    ~I() { printf("moon\n"); }
    int Get() { return i; }
    void operator+=(const I& op) { i+=op.i; }
};
void f(I& x, I y) {
    y += 1000;
    x += y;
}
int main() {
    I i1;
    I i2(20);
    i2 += 400;
    f(i1, i2);
    printf("%d %d\n", i1.Get(), i2.Get());
    return 0;
}
```

9. Пусть производный класс получен из базового способом public-наследования и имена добавленных в производном классе членов отличаются от имен базового класса. В чем отличие статуса private и public для членов базового класса в полученном производном классе?

10. Что напечатает следующая программа?

```
class Ex {
    int code;
public:
    Ex(int i) : code(i) {}
    Ex(const Ex& ex) : code(ex.code) {}
    int Get() const { return code; }
};
struct Ex60 : Ex {
    Ex60() : Ex(60) {}
};
void f() {
    throw Ex60();
    printf("sword\n");
}
void t() {
    try { f(); }
    catch(Ex60 &x) {
        printf("lance\n");
        throw Ex(x.Get() + 1);
        printf("dagger\n");
    }
    catch(Ex &) { printf("knife\n"); }
    printf("hammer\n");
}
int main() {
    try { t(); }
    catch(Ex &x) { printf("arche %d\n", x.Get()); }
    catch(...) { printf("pistole\n"); }
    return 0;
}
```

В.2_2010
Ф.И.О._____ гр.____

1	2	3	4	5	6	7	8	9	10

1. Описать класс `B` таким образом, чтобы все конструкции функции `main()` были верными, а на экран выдалось `10 20 30`.

```
int main () {
    B b1,  b2 = b1, b3 (b2);
    cout << b1.get() << ' ' << b2.get() << ' ' << b3.get() << endl;
    return 0;
}
```

Изменять функцию `main`, использовать механизм исключений, а также функции `exit()`, `terminate()` и т. п. для досрочного завершения программы **запрещается**.

2. Если в реализации функций `D::h()` и `main()` есть ошибки, объясните, в чем они заключаются. В теле функций `main()` и `D::h()` к именам каждой правильной конструкции добавьте с помощью символа `::` префикс, указывающий, с какой областью видимости связано имя.

```
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. Дан текст описания двух структур:

```
struct S {
   virtual void f() const = 0;
   virtual ~S() {}
};
struct A {
   S s;
   int i;
};
```

Есть ли в этом тексте ошибки? Если да, то в чём они заключаются?

4. Есть ли ошибки в следующих заголовках шаблонов? Если есть, поясните, в чём они заключаются.

```
template <int n> class A { /*...*/ };
template <float f> class B { /*...*/ };
template <class Cs> class C { /*...*/ };
```

5. Что напечатает следующая программа?

```
class A {
    int i;
public:
    A(int x) { i = x; printf("mercury\n"); }
    virtual ~A() { printf("venus\n"); }
    int f() const { return 96; }
    virtual int g() const { return i; }
    int h() const { return i - f() - g(); }
};
class B : public A {
public:
    B(int x) : A(x+20) { printf("earth\n"); }
    ~B() { printf("mars\n"); }
    int f() const { return 8; }
    virtual int g() const { return 3; }
    int h() const { return f() + g(); }
};
int main() {
    B b(17);
    A* p = &b;
    printf("result = (%d ; %d)\n", p->h(), b.h());
    return 0;
}
```

6. Описать класс C таким образом, чтобы были верными все конструкции следующего фрагмента программы:

```
const char C::c = '+';
int main () {
    C ob;
    C::f();
    return 0;
}
```

7. Даны описания структуры, переменной и функции:

```
struct mystr {
    int a, b;
};
int i = sizeof(mystr);
int f(mystr s) {
    return 0;
}
```

Дополните описание структуры `mystr` (не изменяя описание функции `f`) так, чтобы только описание `f` стало ошибочным.

8. Что напечатает следующая программа?

```
class I {
    int i;
public:
    I() : i(6) { printf("owl\n"); }
    I(int a) : i(a) { printf("sheep %d\n", i); }
    I(const I& other) : i(other.i) { printf("horse %d\n", i); }
    ~I() { printf("wolf\n"); }
    int Get() { return i; }
    void operator*=(const I& op) { i*=op.i; }
};
void f(I x, I &y) {
    x *= 1;
    y *= x;
}
int main() {
    I i1;
    I i2(3);
    i1 *= 7;
    f(i1, i2);
    printf("%d %d\n", i1.Get(), i2.Get());
    return 0;
}
```

9. ООП. Что такое полиморфизм? Какие виды полиморфизма реализованы в C++, с помощью каких механизмов? Когда эти механизмы работают, во время выполнения программы или во время работы компилятора?

10. Что напечатает следующая программа?

```
class Ex {
    int code;
public:
    Ex(int i) : code(i) {}
    Ex(const Ex& ex) : code(ex.code) {}
    int Get() const { return code; }
};
struct Ex51 : Ex {
    Ex51() : Ex(51) {}
};
void f() {
    throw Ex51();
    printf("train\n");
}
void t() {
    try { f(); }
    catch(Ex51 &x) {
        printf("plane\n");
        throw Ex(x.Get() + 1);
        printf("helicopter\n");
    }
    catch(Ex &) { printf("car\n"); }
    printf("truck\n");
}
int main() {
    try { t(); }
    catch(Ex &x) { printf("boat %d\n", x.Get()); }
    catch(...) { printf("rocket\n"); }
    return 0;
}
```

В.3_2010
Ф.И.О._____ гр.____

1	2	3	4	5	6	7	8	9	10

1. Описать класс C таким образом, чтобы все конструкции функции `main()` были верными, а на экран выдалось `14 10 48` .

```
int main () {
    C c1(7), c2 = 5, c3(c1+c2);
    cout << c1.get() << ' ' << c2.get() << ' ' << c3.get() << endl;
    return 0;
}
```

Изменять функцию `main`, использовать механизм исключений, а также функции `exit()`, `terminate()` и т. п. для досрочного завершения программы **запрещается**.

2. Если в реализации функций `S::g()` и `main()` есть ошибки, объясните, в чем они заключаются. В теле функций `main()` и `S::g()` к именам каждой правильной конструкции добавьте с помощью символа `::` префикс, указывающий, с какой областью видимости связано имя.

```
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);
        f(-1 , 1);
        y = 2;
    }
};
int main () {
    S b;
    f(5);
    f('+', 6);
    return 0;
}
```

3. Дан текст фрагмента программы:

```
class B {
public:
   virtual int f() = 0;
   int g() { return f() * 10; }
   virtual ~B() {}
};
int h(B b) { return b.g() + 2; }
```

Есть ли в этом тексте ошибки? Если да, то в чём они заключаются?

4. Есть ли ошибки в следующих заголовках шаблонов? Если есть, поясните, в чём они заключаются.

```
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. Что напечатает следующая программа?

```
class A {
    int i;
public:
    A(int x) { i = x; printf("dog\n"); }
    virtual ~A() { printf("cat\n"); }
    int f() const { return i + g() + h(); }
    virtual int g() const { return i; }
    int h() const { return 5; }
};
class B : public A {
public:
    B() : A(21) { printf("sheep\n"); }
    ~B() { printf("horse\n"); }
    int f() const { return g() - 3; }
    virtual int g() const { return 7; }
    int h() const { return 9; }
};
int main() {
    B b;
    A* p = &b;
    printf("result = (%d ; %d)\n", p->f(), b.f());
    return 0;
}
```

6. Описать класс X таким образом, чтобы были верными все конструкции следующего фрагмента программы:

```
int main () {
    const X x;
    X::g();
    x.h();
    return 0;
}
```

7. Опишите структуру с именем `smartstr`, удовлетворяющую двум условиям:
(1) можно создать объект типа smartstr;
(2) нельзя создать массив элементов типа smartstr в динамической памяти.

8. Что напечатает следующая программа?

```
class I {
    int i;
public:
    I() : i(5) { printf("fist\n"); }
    I(int a) : i(a) { printf("lance %d\n", i); }
    I(const I& other) : i(other.i) { printf("dagger %d\n", i); }
    ~I() { printf("pistole\n"); }
    int Get() { return i; }
    void operator+=(const I& op) { i+=op.i; }
};
void f(I& x, I y) {
    y += 1000;
    x += y;
}
int main() {
    I i1;
    I i2(30);
    i2 += 700;
    f(i1, i2);
    printf("%d %d\n", i1.Get(), i2.Get());
    return 0;
}
```

9. ООП. Перечислите основные постулаты (механизмы) ООП. Каким образом реализация каждого постулата упрощает процесс создания сложного ПП?

10. Что напечатает следующая программа?

```
class Ex {
    int code;
public:
    Ex(int i) : code(i) {}
    Ex(const Ex& ex) : code(ex.code) {}
    int Get() const { return code; }
};
struct Ex90 : Ex {
    Ex90() : Ex(90) {}
};
void f() {
    throw Ex90();
    printf("beta\n");
}
void t() {
    try { f(); }
    catch(Ex90 &x) {
        printf("gamma\n");
        throw Ex(x.Get() + 1);
        printf("delta\n");
    }
    catch(Ex &) { printf("epsilon\n"); }
    printf("omicron\n");
}
int main() {
    try { t(); }
    catch(Ex &x) { printf("tau %d\n", x.Get()); }
    catch(...) { printf("omega\n"); }
    return 0;
}
```

В.4_2010
Ф.И.О._____ гр.____

1	2	3	4	5	6	7	8	9	10

1. Описать класс D таким образом, чтобы все конструкции функции `main()` были верными, а на экран выдалось `30 10 20`.

```
int main() {
    D d1, d2 (1), d3(2);
    d1 = d2 + d3;
    cout << d1.get() << ' ' << d2.get() << ' ' << d3.get() << endl;
    return 0;
}
```

Изменять функцию `main`, использовать механизм исключений, а также функции `exit()`, `terminate()` и т. п. для досрочного завершения программы **запрещается**.

2. Если в реализации функций `X::g()` и `main()` есть ошибки, объясните, в чем они заключаются. В теле функций `main()` и `X::g()` к именам каждой правильной конструкции добавьте с помощью символа `::` префикс, указывающий, с какой областью видимости связано имя.

```
void f(){ putchar ('f'); }
class Y {
public:
    void f() {}
};
class X : public Y {
    double a;
    X(int k = 0) { a = k; }
public:
    X(double r ) { a = r; }
    void f(int x) { a = x; }
    void f(int x, int y = 5) { a = x - y; }
    void g() {
        f();
        f(3);
        f(1, 2);
    }
};
int main () {
    Y y;
    X a;
    X b(2.5);
    b.f();
    return 0;
}
```

3. Дан фрагмент программы:

```
struct MM {
   int i;
   virtual int f() const = 0;
   virtual ~MM() {}
};
struct NN : public MM {
   int j;
   int g() const { return j; }
};
NN nn1;
```

Есть ли в этом тексте ошибки? Если да, то в чём они заключаются?

4. Есть ли ошибки в следующих заголовках шаблонов? Если есть, поясните, в чём они заключаются.

```
class myclass { public: myclass() {} };
template <int n> class A { /*...*/ };
template <class Cs> class C { /*...*/ };
template <myclass c> class B { /*...*/ };
```

5. Что напечатает следующая программа?

```
class A {
    int i;
public:
    A(int x) { i = x; printf("sword\n"); }
    virtual ~A() { printf("lance\n"); }
    int f() const { return 49; }
    virtual int g() const { return i; }
    int h() const { return i - f() - g(); }
};
class B : public A {
public:
    B(int x) : A(x+60) { printf("dagger\n"); }
    ~B() { printf("knife\n"); }
    int f() const { return 5; }
    virtual int g() const { return 2; }
    int h() const { return f() + g(); }
};
int main() {
    B b(13);
    A* p = &b;
    printf("result = (%d ; %d)\n", p->h(), b.h());
    return 0;
}
```

6. Описать класс B таким образом, чтобы были верными все конструкции следующего фрагмента программы:

```
char * B::s = "ABC";
int main () {
    const B b;
    cout << B::s << b.h() << endl;
    return 0;
}
```

7. Выпишите профили (т.е. заголовки) всех методов (включая неявные), определённых для следующей структуры:

```
struct str1 {
   int a, b;
   str1(int m) : a(m), b(m) {}
};
```

8. Что напечатает следующая программа?

```
class I {
    int i;
public:
    I() : i(9) { printf("pedestrian\n"); }
    I(int a) : i(a) { printf("helicopter %d\n", i); }
    I(const I& other) : i(other.i) { printf("car %d\n", i); }
    ~I() { printf("rocket\n"); }
    int Get() { return i; }
    void operator*=(const I& op) { i*=op.i; }
};
void f(I x, I &y) {
    x *= 1;
    y *= x;
}
int main() {
    I i1;
    I i2(2);
    i1 *= 3;
    f(i1, i2);
    printf("%d %d\n", i1.Get(), i2.Get());
    return 0;
}
```