1. Tekintsük a következő programrészletet. Melyik függvényhívás helyes az alábbiak közül?

```
void fv(int i)
{
        // ...
}

void g(int& i)
{
        // ...
}

const int i = 10;
double d;
```

a. g(d);
b. fv(&i);
c. fv(i);
d. g(&i);

2. Melyik állítás igaz az alábbiak közül?
    a. A char típus előjelessége fordító-függő.
    b. A char lebegőpontos típus.
    c. A char előjel (unsigned) nélküli.
    d. A char előjeles (signed) típus.

3. Melyik azonosító szabályos a C++ szabályai szerint?
    a. X::f
    b. int
    c. std
    d. vector<int>

4. Milyen destruktora(i) van(nak) az alábbi struct-nak?

```
struct X
{
        X(int i) { ... }
};
```

a. csak egy int paraméteres destruktora
b. virtuális destruktora
c. struck-nak nem lehet destruktora, ha írunk neki fordítási hibát kapunk.
d. Egy automatikusan generált destruktora van, ami nem csinál semmit.

5. Melyik konstrukciót nem a C++11 vezette be?
    a. funktor
    b. nullptr
    c. decltype
    d. Lambda függvények

6. Mi a típusa a ”szia” + 2 kifejezésnek?
    a. const char*
    b. const char[7]
    c. const char[5]
    d. const char[6]

7. Melyik kulcsszó nem a tárolási osztályt specifikálja egy deklarációban ill. definícióban?
    a. extern
    b. int
    c. static
    d. auto

8. Az alábbiak közül melyik fut le az objektum élettartama végén?
    a. delete operátor
    b. dispose tagfüggvény
    c. az unset függvény
    d. destruktor

9. Melyik konténer szekvenciális?
    a. std::map
    b. std::unordered_multiset
    c. std::list
    d. std::set

10. Az alábbi kódban a csillagozott helyen mi a this-nek a típusa?

```
struct Person
{
        std::string get_name() const
        {
                // (*)
        }
};
```

    a. Person&
    b. const Person*
    c. Person*
    d. std::string

11. Melyik memóriaallokáció helyes az alábbiak közül?
    a. char* a = new char*[20];
    b. char a = new char(20.0);
    c. char* a = new char[20];
    d. char a = new char[20];

12. Mi lesz a b változó értéke az alábbi kód lefutása után?

```
int j = 0;
int* p = &j;
bool b = p;
```

    a. true
    b. implementáció-függő
    c. false
    d. fordítási hiba

13. Melyik deklarációra illeszkedik a csillaggal jelölt sorban meghívott művelet?

```
class Foo
{
        // ...
};

Foo f;
Foo g = f;   // (*)
```

    a. void Foo::operator()();
    b. Foo::Foo(const Foo& rhs);
    c. Foo::Foo();
    d. Foo& Foo::operator=(const Foo& rhs);

14. Az alábbi példában a **Foo f(10);** konstruktor hívása után mennyi lesz **f.x** értéke?

```
struct Foo
{
        int x, y;
        Foo(int i) : y(i), x(y++) { }
};
```

    a. 11
    b. nemdefiniált
    c. 0
    d. 10

15. Az alábbi kód alapján melyik állítás igaz az alábbiak közül?

```
struct Base
{
        virtual ~Base() { }
};

struct Der: public Base
{
        virtual ~Der() { }
};

Base* p = new Der();
delete p;
```

a. A delete hatására először a Der destruktora azután a Base destruktora hívódik meg.
b. A delete hatására csak a Base destruktora hívódik meg.
c. A delete hatására csak a Der destruktora hívódik meg.
d. A delete hatására először a Base destruktora azután a Der destruktora hívódik meg.

16. Mit ír ki az alábbi program?

```
void f(int a, int b)
{
        std::cout << a << b;
}

int i = 1;
f(i++, i++);
```

a. a fenti esetben nem definiált, hogy 12-t vagy 21-et ír ki, mert nincs szekvenciapont
b. 11
c. 22
d. 21

17. Melyik funktor típus az alábbiak közül?
    a. std::deque<int>
    b. double*
    c. std::less<double>
    d. std::pair<std::string, std::string>

18. Az alábbi típusok közül melyik polimorfikus?

```
Struct X;

Struct A
{
        A(const X& b);
        A(int i);
}

struct B
{
  static const int a;
};

struct Base{};
struct C : public Base
{};

struct D
{
  virtual ~D();
};
```

a. D
b. C
c. B
d. A