A C++ kódokban lévő makrókat melyik egység dolgozza fel az alábbiak közül?
preprocessor
A szabványos C++-ban nem is írhatunk makrókat (csak C-ben)
assembler
linker

Az alábbi példában a Foo f(5); konstruktor hívása után mennyi lesz f.b értéke?
struct Foo
{
int a, b;
Foo(int c):a(c*2),b(c*3) {}
};

15
nemdefiniált
0
10

Az alábbiak közül melyiket kötelező inicializálni az inicializáló listában?
az összes adattagot
a referenciákat
az STL-es konténereket
a pointereket

Adott egy tipus, melynek mérete nem egyezik meg a típus adattagjai méretének összegével. Mi történhetett?
Megörököltük annak az osztálynak a tagjait is, amelyik minden C++ osztálynak az őse.
Megfeledkeztünk a header guard-okról és több helyre is be include-oltuk a header filet.
Találtunk egy bugot a fordítóprogramban.
A fordítóprogram szóhatárra optimalizálta az adattago(ka)t

Definiálhatunk-e egy C++ függvény legbelső blokkjában két azonos nevű változót?
Igen, definiálhatunk.
Nem.
Ezt csak a g++ fordítóprogram támogatja.
Csak akkor, ha különböző a típusuk.

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++) {}
};
0
10
nemdefiniált
11

Adott egy tipus, melynek mérete nem egyezik meg a típus adattagjai méretének összegével. Mi történhetett?
A this pointer miatt nagyobb az osztály mérete.
Megörököltük annak az osztálynak a tagjait is, amelyik minden C++ osztálynak az őse.
Az osztályunknak van virtuális függvénye, így létrejött a virtuális tábla pointer.
Találtunk egy bugot a fordítóprogramban.

Az alábbiak közül melyiket kötelező inicializálni az inicializáló listában?
az összes adattagot
tömböket
semmit sem kötelező inicializálni
a konstansokat

Adott az alábbi X típus és f függvény. Az f(x) függvény hívásakor az X típus melyik műveletét hajtjuk végre a paraméter átadásához?

```
struct X
{
// ...
};

void f(X a)
{
// ...
}
X x;
```

Az X típus default konstruktorát.
Nem hajtunk végre műveletet, mert x hivatkozás szerint adódik át.
Az X típus értékadó operátorát.
Az X típus copy konstruktorát.

Az alábbi függvény deklarációk alapján melyik tagfüggvény hívható meg const Foo objektumon?

```
struct Foo
{
const int a(int i);
int b(const int i);
virtual int c(int i);
int d(int i) const;
Foo() {}
};
const Foo foo;
```

foo.d(2);
foo.c(0);
foo.a(3);
foo.b(12);

Az std::sort algoritmus melyik kontÄŠnerrel használható?

std::set
std::list
std::auto_ptr
std::deque

Az alábbi függvény deklarációk alapján melyik tagfüggvény hívható meg const Foo objektumon?

```
struct Foo
{
virtual void a(const int i);
const int& b(const int i);
void c() const;
const Foo& d(const Foo& f);
Foo() {}
};
const Foo foo;
```

foo.d(foo);
foo.a(3);
foo.c()
foo.b(12);

Az X::f() függvényhívás során mit ír ki a program?

```
int i = 1;
namespace X
{
int i = 2;

void f()
{
int a = i + 1;
int i = ::i - 1;
std::cout << a << ", " << i << std::endl;
```

}
}
semmit, fordítási hiba keletkezik
2, 1
3, 2
3, 0

Az alábbi kódban a csillagozott helyen mi this-nek a típusa?
struct Foo
{
void f()
{
// (*)
}
};

Foo*
void*
Foo&
const Foo*

Adott az alábbi X típus és f függvény. Az f(x) függvény hívásakor az X típus melyik műveletét hajtjuk végre a paraméter átadásához?
class X
{
// ...
};

void f(X a)
{
// ...
}

X x;
Az X típus értékadó operátorát.
Nem hajtunk végre műveletet, mert x hivatkozás szerint adódik át.
Az X típus copy konstruktorát.
Az X típus default konstruktorát.

Melyik állítás igaz az alábbiak közül?
A bázisosztály konstruktorai nem öröklődnek a származtatott típusba.
A konstruktor közül csak a copy konstruktor lehet virtuális, hogy felüldefiniálható legyen a másolás.
Polimorf osztályok esetében az összes konstruktornak virtuálisnak kell lennie.
Nem lehet olyan osztályból származtatni, amelynek nincsen virtuális destruktora.

Mi nem lehet template paraméter az alábbiak közül?
Típus
Külső szerkesztésű objektum címe
Logikai konstans
Lebegőpontos konstans

Melyik állítás igaz az alábbiak közül?
Nem lehet származtatni typedef által meghatározott típusból.
Az objektumok dinamikus típusát ismeri a fordítóprogram.
Paraméterdedukció csak függvények esetében használható.
A paraméterdedukció futási időben történik.

Melyik állítás igaz az alábbiak közül?
A long long típust 8 byte-on ábrázolja a C++.
A sizeof(long int)<= sizeof(long long) reláció mindig igaz.
A szabványos C++ nem definiálja a long long típust.
A sizeof(long double) == sizeof(long long) reláció mindig igaz.

Mennyi a 012 konstans értéke?
18
12
0.12
10

Melyik azonosító szabályos a C++ szabályai szerint?
101_kiskutya
_1
miez?
jo!

Melyik kulcsszó nem a tárolási osztályt specifikálja egy deklarációban ill. definícióban?
register
static
public
auto

Melyik igaz az alábbiak közül?
A dinamikus változók a stack-en jönnek létre.
A dinamikus változók a statikus tárterületen jönnek létre.
A dinamikus változók a heap-en jönnek létre.
A dinamikus változók a winchester-en jönnek létre.

Mennyi lesz foo.a értéke?

```
struct Foo
{
int a;

Foo(int i)
{
Foo(i, 0);
}

Foo(int i, int j)
{
a = j;
}
};
Foo foo(4);
```

4
0
Nemdefiniált
Fordítási hibát kapunk.

Mennyi lesz foo.a értéke?

```
struct Foo
{
int a;

Foo(int i):Foo(i, 0)
{
}

Foo(int i, int j):a(i)
{
}
};

Foo foo(4);
```

Nemdefiniált
4
0
Fordítási hibát kapunk.

Melyik igaz az alábbiak közül?
template
class Foo;

int i;

template
void f(const T& t)
{
typename Foo::N * i;
// ...
}
A fordítóprogram a fenti kódot úgy elemzi tovább, hogy a függvény sablon első sorában egy i nevű pointerrel elfedtük a globális int i-t.
A fordítóprogram a fenti kódot úgy elemzi tovább, hogy végeztünk egy szorzást a függvény sablon első sorában.
Nem fedhetjük el a külső i azonosítót, ezért a fenti kód fordításakor hiba keletkezik.
A fordítóprogramtól függ, hogy a fenti kódban szorzást végzünk vagy egy pointert hozunk létre.

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

Melyik kontĂŠner asszociatív?
std::hashmap
std::set
std::vector
std::list

Mit nevezünk funktornak?
Azokat az alprogramokat, amelyeknek nem void a visszatérési érték típusa.
Implementáció függő.
Azokat az objektumokat, amelyek van operator()-a.
Azokat az alprogramokat, amelyeknek void a visszatérési érték típusa.

Melyik nyelvi konstrukció támogatja párhuzamos programok írását C++-ban?
polimorfizmus
template
Nincs olyan nyelvi konstrukció, ami támogatja párhuzamos programok írását.
protected

Mennyi a 012 konstans értéke?
0.12
10
18
12

Melyik azonosító szabályos a C++ szabályai szerint?
std::stack
vector
t[i]
~dtor

Mitől válik egy osztály absztrakttá?
Van bázisosztálya
A tagfüggvényeinek csak a deklarációja ismert.
Van virtuális destruktora.
Van tisztán virtuális tagfüggvénye.

Melyik vezet fordítási hibához az alábbi osztály template definíciók közül?

```
template
class A
{
};
template
class B
{
};
template
class C
{
};
template
class D
{
};
```

D
B
A
C

Melyik reláció hamis az alábbiak közül?
sizeof(double) < sizeof(long double)
sizeof(unsigned char) == sizeof(char)
sizeof(short)<=sizeof(int)
sizeof(float)<=sizeof(double)

Melyik reláció igaz az alábbiak közül?
sizeof(float) <= sizeof(double)
sizeof(int) <= sizeof(char)
sizeof(unsigned char) < sizeof(char)
sizeof(bool) < sizeof(char)

Melyik nem definíció az alábbiak közül?
int k;
const int l = 1;
static int i;
extern int j;

Melyik igaz az alábbiak közül?
A dinamikus változók a winchester-en jönnek létre.
A dinamikus változók a statikus tárterületen jönnek létre.
A dinamikus változók a stack-en jönnek létre.
A dinamikus változók a heap-en jönnek létre.

Melyik reláció hamis az alábbiak közül?
sizeof(bool) == sizeof(char)
sizeof(char) == sizeof(signed char)
sizeof(float)<=sizeof(long double)
sizeof(short)<=sizeof(long int)

Mennyi a 0x11 konstans értéke?
9
17
3
11

Melyik nem preprocesszor direktíva?
#elseif
#define
#else
#elif

Melyik nem definíció az alábbiak közül?
struct Foo { // ... };
int i;
class Foo { // ... };
void f(int i);

Melyik igaz az alábbiak közül?
A globális változók a stack-en jönnek létre.
A globális változók a statikus tárterületen jönnek létre.
A globális változók a heap-en jönnek létre.
A globális változók a winchester-en jönnek létre.

Milyen konstruktora(i) van(nak) az alábbi struct-nak?
struct X
{
X(int) {...}
};

csak default konstruktora
csak copy konstruktora
copy konstruktora es egy int paraméteres konstruktora
csak egy int paraméteres konstruktora

Mit ír ki a képernyőre az alábbi kódrészlet?
template
const T& max(const T& a, const T& b)
{
return a>b?a:b;
}
std::cout << max("abc", "sef");

sef
abc
Nemdefiniált az eredménye
Fordítási hiba keletkezik.

Melyik típusnak van push_front tagfüggvénye?
std::vector
std::set
std::stack
std::deque

Hány byte-on tárol a C++ egy double-t?
implementáció-függő
8
4
6

Mit ír ki a képernyőre az alábbi kódrészlet?
template
const T& max(const T& a, const T& b)
{
return a>b?a:b;
}
std::cout << max("abc", max("def", "xyz"));

abc
def
xyz
Nemdefiniált az eredménye

Melyik állítás igaz az alábbiak közül?
Nem lehet olyan programot írni C++-ban, amelyik adatbázisszerverhez kapcsolódna.
Lehet olyan programot írni C++-ban, amelyik fordítása közben algoritmusokat hajt végre.
Nem lehet párhuzamos programot írni C++-ban.
Lehet olyan programot írni C++-ban, amelyik fordítás nélkül is futhat.

Melyik állítás igaz az alábbiak közül?
A 4e-1f és a 4.1 konstansok típusa megegyezik.
A 4e-1 és a 0.4 konstansok értéke megegyezik.
A 4e2 és a 4.2L konstansok típusa megegyezik.
A 4e-1f és a 4.1 konstansok értéke megegyezik.

Melyik azonosító szabályos a C++ szabályai szerint?
_18
ures-e
!b
1001_ejszaka

Mi a csilagozott sorban meghívott művelet neve?
```
class Foo
{
...
};
Foo f;
Foo g = f; // (*)
```
copy konstruktor
értékadó operátor
default konstruktor
destruktor

Melyik állítás igaz az alábbiak közül?
Absztrakt osztálynak nem lehet konstruktora.
Absztrakt osztálynak nem lehet adattagja.
Absztrakt osztálynak nem lehet objektumot létrehozni.
Absztrakt osztálynak nem lehet származtatni.

Mi a típusa a 5e2f literálnak?
ez nem szabályos konstans
float
double
int

Melyik állítás igaz az alábbiak közül?
A dynamic_cast használatához nem lehet statikus adattagja az osztálynak.
A dynamic_cast használatához polimorf osztályokra van szükség.
A dynamic_cast soha nem dob kivételt.
A dynamic_cast fordítás idejű típuskonverziót végez.

Melyik állítás igaz az alábbiak közül?
Nem lehet sablon (template) tagfüggvénye egy nem-template osztálynak.
A struct konstrukcióból nem lehet sablont (template-t) írni.
Az enum konstrukcióból lehet sablont (template-t) írni.
A typedef konstrukcióból nem lehet sablont (template-t) írni.

Mi a típusa a 5f2e konstansnak?
double
ez nem szabályos konstans
int
float

Melyik igaz az alábbiak közül?
```
struct X
{
X(int i = 0) {}
};
```
A fenti struct-nak nincs default konstruktora.
A fenti struct-nak nincs copy konstruktora.
A fenti struct-nak csak default konstruktora van.
A fenti struct-nak van default konstruktora.

Melyik konténer szekvenciális?
std::deque
std::set
std::arraylist
std::map

Melyik állítás igaz az alábbiak közül?
A sizeof(int) == sizeof(const int*) reláció mindig igaz.
Egy const int* típusú pointer mutathat változóra.
Egy const int* típusú pointer mérete 4 byte.
Egy const int* típusú pointer megváltoztathatja a mutatott értéket.

Melyik reláció hamis az alábbiak közül?
sizeof(short)<=sizeof(long int)
sizeof(float)<=sizeof(long double)
sizeof(char) == sizeof(signed char)
sizeof(bool) == sizeof(char)

Mennyi a 018 konstans értéke?
Nincs ilyen konstans
0.18
24
18

Melyik nem preprocesszor direktíva?
#else
#while
#elif
#undef

Mi lesz az a változó értéke a függvényhívás után?
int a = 1, b = 2;
void f(int* x, int* y)
{
int t = *x;
*x = *y;
*y = t;
}
f(a,b);
nem definiált
1
semmi, fordítási hiba keletkezik
2

Melyik állítás igaz egy konstans objektum esetében?
Az objektumnak csak private adattagja lehet.
Az objektumnak csak a konstans tagfüggvényei hívhatóak meg.
Az objektumnak csak azok a tagfüggvényei hívhatóak meg, amelyek nem módosítják az adattagjait.
Az objektum csak default konstruktorral hozható létre.

Mi történik az alábbi függvényhíváskor?
template <typename T>
T max(const T& a, const T& b);
max(4.3, 23);

Fordítási hiba keletkezik
Futás idejű hiba keletkezik
Mindkét paraméter int-té konvertálódik
Mindkét paraméter double-lé konvertálódik

Melyik állítás igaz az alábbiak közül?
Nem származtathatunk az std::string típusból, mert nincs virtuális destruktora.
Nem származtathatunk az std::string típusból, mert az nem típus, hanem typedef.
Származtathatunk az std::string típusból.
Nem származtathatunk az std::string típusból, mert nincsenek protected adattagjai.

Melyik igaz az alábbiak közül?
template
class Foo;
int i;
template
void f(const T& t)
{
Foo::N * i;
// ...
}

A fordítóprogram a fenti kódot úgy elemzi tovább, hogy a függvény sablon első sorában egy i nevű pointerrel elfedtük a globális int i-t.
A fenti kód nem fordul le, mert nem írtuk ki a typename kulcsszót.
A fordítóprogram a fenti kódot úgy elemzi tovább, hogy végeztünk egy szorzást a függvény sablon első sorában.
A fordítóprogramtól függ, hogy a fenti kódban szorzást végzünk vagy egy pointert hozunk létre.

Melyik értékadás szabályos az alábbi kód után?
int i = 10;
const int j = 15;
const int *p = &j;
p = &i;
*p = i;
p *= i;
p = *j;

Mi a típusa a 0xff konstansnak?
char*
double
int
double*

Melyik nem definíció az alábbiak közül?
int k;
static int i;
extern int j;
const int l = 1;

Mi lesz az a változó értéke a függvényhívás után?
int a = 1, b = 2;
void f(int& x, int& y)
{
int t = x;
x = y;
y = t;
}
f(a,b);

nem definiált
semmi, fordítási hiba keletkezik
2
1

Melyik kódrészlet helyes?
struct Foo { template <bool f> void bar() const { // ... } }; Foo f; f.bar<true>();
template <int N> enum A { Elem = N };
template <typename T> typedef std::set<T, std::greater<T> > GreaterSet;
template <typename T = int> const T& max(const T& a, const T& b);

Mikor nevezünk erősen típusosnak egy nyelvet?
Erősen típusos, ha a forditóprogram ellenőrzi, hogy definiált-e egy objektum vagy alprogram.
Erősen típusos, ha a futási időben nem keletkezik kivétel.
Erősen típusos, ha minden kifejezés és részkifejezés típusa fordítási időben meghatározott.
Erősen típusos, ha minden kifejezés és részkifejezés típusa futási időben meghatározott.

Melyik igaz az alábbiak közül?
A friend kulcsszóval meghatározhatjuk a közelebbi osztályt többszörös öröklődés esetében.
A friend kulcsszó több osztály logikai csoportosítására szolgál.
Egy friend template osztály esetén példányosításkor nem kötelező explicit megadni a template paramétereket.
Egy friend függvény hozzáférhet az osztály private tagjaihoz.

Milyen konstruktorok hívhatóak az alábbi struct esetében?

```
struct X
{
};
```

copy és default konstruktor
nincsen konstruktora
csak default konstruktor
csak copy konstruktor

Mi a problemája a preprocesszor használatának?
A preprocesszor implementáció-specifikus.
Jelentősen növeli a futási időt.
A Java programozási nyelv nem támogatja, ezért nem tudjuk együtt használni C++-t a Java-val.
Független a C++ nyelvtől, ezért nincs tekintettel a nyelvi szabályokra.

Melyik nem definíció az alábbiak közül?
class Foo { // ... };
struct Foo { // ... };
void f(int i);
int i;

Melyik definíció az alábbiak közül?
void* p;
int f();
extern int i;
struct X;

Melyik paradigma alapján épül fel a C++ Standard Template Library?
funkcionális
generikus
objektum-orientált
iterator

Mi lesz az a változó értéke a függvényhívás után?

```
int a = 1, b = 2;
void f(int x, int y)
{
int t = x;
x = y;
y = t;
}
f(a,b);
```

nem definiált
2
1
semmi, fordítási hiba keletkezik

Melyik állítás igaz az alábbiak közül?
A postfix operator++ mindig a megnövelt értéket adja vissza.
A postfix operator++ mindig hatékonyabb, mint a prefix.
Deklarációban egy plusz paraméterrel tudjuk megkülönböztetni a postfix operator++-t a prefix-től.
Az alaptípusok prefix operator++-nak void a visszatérési érték típusa.

Melyik állítás igaz az alábbiak közül?
Egy int* const típusú pointer nem változtathatja meg a mutatott értéket.
A sizeof(int) == sizeof(int* const) reláció mindig igaz.
Egy int* const típusú pointer mérete 8 byte.
Egy int* const típusú pointer mutathat változóra.

Melyik definíció az alábbiak közül?
extern int i;
struct X;
void* p;
int f();

Melyik állítás igaz az alábbiak közül?
Nem lehet alkalmazni a többszörös öröklődést, ha azonosító ütközés lépne fel.
A C++ tiltja a többszörös öröklődést.
Csak akkor használható a többszörös öröklődés, ha az összes bázisosztálynak van virtuális destruktora.
A C++ engedélyezi a többszörös öröklődést.

Hány byte-on tárol a C++ egy int-et?
1
implementáció-függő
4
8

Hány byte-on tárol a C++ egy short int-et?
8
1
2
implementáció-függő

Lehet-e egy C++ függvényben két azonos nevű változó?
Nem lehet.
Csak akkor, ha különböző blokkban definiálták.
Csak akkor, ha különböző a típusuk.
Csak akkor, ha a láthatóságuk nem esik egybe.

Projektünkben az összes fordítási egység lefordult, de nem jön létre a futtható állomány a build folyamat végén. Mi lehet a baj?
A build folyamat közben nem találtuk meg a preprocessor-t.
A linker nem talált meg egy dinamikus linkelésű library-t.
A linker nem talált meg egy statikus linkelésű library-t.
A virtuális destruktorok hiánya okozta.

Hány byte-on tárol a C++ egy float-ot?
8
implementáció-függő
4
6

Hány byte-on tárol a C++ egy karaktert (char)?
implementáció-függő
1
8
4