**Exercitiul 12**

////////////////////////////////////

//// 1

////////////////////////////////////

#include <iostream>

using namespace std;

class A {

public:

A () {cout << "A";}

~A () {cout << "~A";}

};

class B: public A {

public:

B () {cout << "B";}

~B () {cout << "~B";}

};

class C: public B {

public:

C () {cout << "C";}

~C () {cout << "~C";}

};

int main () {

A \*pa = new C(); delete pa;

return 0;

}

////////////////////////////////////

//// 2

////////////////////////////////////

#include <iostream>

using namespace std;

class A {

public:

A () {cout << "A";}

~A () {cout << "~A";}

};

class B: A {

public:

B () {cout << "B";}

~B () {cout << "~B";}

};

class C: public B {

public:

C () {cout << "C";}

~C () {cout << "~C";}

};

int main () {

A \*pa = new C(); delete pa;

return 0;

}

////////////////////////////////////

//// 3

///////////////////////////////////

#include <iostream>

using namespace std;

class A {

public:

A () {cout << "A";}

virtual ~A () {cout << "~A";}

};

class B: public A {

public:

B () {cout << "B";}

~B () {cout << "~B";}

};

class C: public B {

public:

C () {cout << "C";}

~C () {cout << "~C";}

};

int main () {

A \*pa = new C(); delete pa;

return 0;

}

////////////////////////////////////

//// 4

///////////////////////////////////

#include <iostream>

using namespace std;

class A {

public:

A () {cout << "A";}

~A () {cout << "~A";}

};

class B: public A {

public:

B () {cout << "B";}

~B () {cout << "~B";}

};

class C: public A, public B {

public:

C () {cout << "C";}

~C () {cout << "~C";}

};

int main () {

A \*pa = new C(); delete pa;

return 0;

}

////////////////////////////////////

//// 5

///////////////////////////////////

#include <iostream>

using namespace std;

class A {

public:

A () {cout << "A";}

~A () {cout << "~A";}

};

class B: public A {

public:

B () {cout << "B";}

virtual ~B () {cout << "~B";}

};

class C: public A, public B {

public:

C () {cout << "C";}

~C () {cout << "~C";}

};

int main () {

B \*pb = new C(); delete pb;

return 0;

}

////////////////////////////////////

//// 1

////////////////////////////////////

// Secventa compileaza si afiseaza ABC~A

// Barem:

// Total: 0.5p

// 1.a 0.1p secventa compileaza si afiseaza ABC~C~B~A sau ABC~B~A

// 1.b 0.25p secventa compileaza si afiseaza ABC~A

// 2 0.25p explicare comportamentului secventei de cod (daca explicatiile sunt partial corecte, ~50%, se acorda jumatate din punctaj)

// OBS! Doar una dintre 1.a sau 1.b poate fi alocata.

////////////////////////////////////

//// 2

////////////////////////////////////

// Secventa nu compileaza din cauza liniei 20 deoarece A este o baza privata a lui C, deci nu se poate face upcasting

// Barem:

// Total: 0.5p

// 1.a 0.1p secventa nu compileaza cu explicatii care nu sunt corecte

// 1.b 0.25p secventa nu compileaza cu explicatii corecte

// 2 0.25p Rezolvarea problemei de compilare prin modificarea liniei 8 in: class B: public A {

// OBS! Doar una dintre 1.a sau 1.b poate fi alocata.

// OBS! Punctajul de punctul 2 se aloca doar daca eroare la compilare este identificata corect.

////////////////////////////////////

//// 3

///////////////////////////////////

// Secventa compileaza si afiseaza ABC~C~B~A

// Barem:

// Total: 0.5p

// 1.a 0.1p secventa compileaza si afiseaza ABC~B~A sau ABC~A

// 1.b 0.25p secventa compileaza si afiseaza ABC~C~B~C

// 2 0.25p explicare comportamentului secventei de cod (daca explicatiile sunt partial corecte, ~50%, se acorda jumatate din punctaj)

// OBS! Doar una dintre 1.a sau 1.b poate fi alocata.

////////////////////////////////////

//// 4

///////////////////////////////////

// Secventa nu compileaza din cauza liniei 20 deoarece A este o baza ambigua pentru C si compilatorul nu stie pe care sa o foloseasca pentru upcasting

// Barem:

// Total: 0.5p

// 1.a 0.1p secventa nu compileaza cu explicatii care nu sunt corecte

// 1.b 0.25p secventa nu compileaza cu explicatii corecte

// 2 0.25p Rezolvarea problemei de compilare prin modificarea liniei 13 in: class C: public B {

// OBS! Doar una dintre 1.a sau 1.b poate fi alocata.

// OBS! Punctajul de punctul 2 se aloca doar daca eroare la compilare este identificata corect.

////////////////////////////////////

//// 5

///////////////////////////////////

// Secventa compileaza si afiseaza AABC~C~B~A~A

// Barem:

// Total: 0.5p

// 1.a 0.1p secventa compileaza si afiseaza ABC~C~B~A sau ABC~B~A sau ABC~A

// 1.b 0.25p secventa compileaza si afiseaza AABC~C~B~A~A

// 2 0.25p explicare comportamentului secventei de cod (daca explicatiile sunt partial corecte, ~50%, se acorda jumatate din punctaj)

// OBS! Doar una dintre 1.a sau 1.b poate fi alocata.