Problema 5:

**Barem:**

**Variantele 1,2:**

Nu si motivatie corecta + modificari corecte: 0.5

Nu si motivatie corecta, dar fara modificari: 0.3

Nu si motivatie aproape corecta: 0.1

**Variantele 3, 4,5:**

Da si afisarile corecte, cu explicatii complete si coerente (despre constructori + apel de functie): 0.5

Da si la explicatii nu se spune nimic despre apelul functiei: 0.3

Da si rezultat aproape corect, cu explicatii: 0.1

**Alte penalizari la explicatii:**

* nu se explica de ce se merge pe functia din A desi se apeleaza un obiect de tip B: penalizare 0.1
* nu se mentioneaza ascunderea functiei fara parametri de catre cea cu parametru (supraincarcarea functiilor virtuale): penalizare 0.1

**Da sau nu fara explicatii nu se puncteaza.**

Varianta 1: NU COMPILEAZA linia 21 da eroare: am.r(8) trebuia fara parametru

#include <iostream>

class A {

protected:

int nm;

public:

A(int hbr = 1) : nm(hbr) { std::cout << "?"; }

int ha() { return nm; }

virtual void r() const {};

virtual ~A() {};

};

class B : public A {

int d;

public:

B(int b = 2) : d(b) { std::cout << "!!"; }

void r(int z) const { std::cout << nm << " " << z << "\n"; }

};

void warranty(const A& am) {

am.r(8);

}

int main() {

A ha;

B un(ha.ha());

warranty(un);

}

Varianta 2: NU COMPILEAZA se incearca instantierea unei clase abstracte

#include <iostream>

class A {

protected:

int nm;

public:

A(int hbr = 1) : nm(hbr) { std::cout << "?"; }

int ha() { return nm; }

virtual void r() const {};

virtual ~A()=0;

};

class B : public A {

int d;

public:

B(int b = 2) : d(b) { std::cout << "!!"; }

void r(int z) const { std::cout << nm << " " << z << "\n"; }

};

void warranty(const A& am) {

am.r();

}

int main() {

A ha;

B un(ha.ha());

warranty(un);

}

Varianta 3: COMPILEAZA si afiseaza ??!!

#include <iostream>

class A {

protected:

int nm;

public:

A(int hbr = 1) : nm(hbr) { std::cout << "?"; }

int ha() { return nm; }

virtual void r() const {};

virtual ~A() {};

};

class B : public A {

int d;

public:

B(int b = 2) : d(b) { std::cout << "!!"; }

void r(int z) const { std::cout << nm << " " << z << "\n"; }

};

void warranty(const A& am) {

am.r();

}

int main() {

A ha;

B un(ha.ha());

warranty(un);

}

Varianta 4: COMPILEAZA si afiseaza ????!

#include <iostream>

class A {

protected:

int nm;

public:

A(int hbr = 1) : nm(hbr) { std::cout << "??"; }

int ha() { return nm; }

virtual void r() const {};

virtual ~A() {};

};

class B : public A {

int d;

public:

B(int b = 2) : d(b) { std::cout << "!"; }

void r(int z) const { std::cout << nm << " " << z << "\n"; }

};

void warranty(const A& am) {

am.r();

}

int main() {

A ha;

B un(ha.ha());

warranty(un);

}

Varianta 5: COMPILEAZA si afiseaza ????! 1 8

#include <iostream>

class A {

protected:

int nm;

public:

A(int hbr = 1) : nm(hbr) { std::cout << "??"; }

int ha() { return nm; }

virtual void r() const {};

virtual ~A() {};

};

class B : public A {

int d;

public:

B(int b = 2) : d(b) { std::cout << "!"; }

void r(int z) const { std::cout << nm << " " << z << "\n"; }

};

void warranty(const B& am) {

am.r(8);

}

int main() {

A ha;

B un(ha.ha());

warranty(un);

}