Anteprima di test

parte teorica

Data: Mon Feb 17 15:55:09 2014 Punteggi massimi: 53

1. Dynamic Binding Java *(8 Punti)*

Date le segenti dichiarazioni:

```
1 class Computer {
2   void setCPU(int l) {
3     System.out.println("C");
4   }
5 }
6 class NoteBook extends Computer {
7   void setCPU(int l) {
8     System.out.println("N");
9   }
10}
11class Tablet extends Computer {
12  void setCPU(short l) {
13    System.out.println("T");
14  }
15}
16...
17Object oc = new Computer();
18Computer cc = new Computer();
19Computer cn = new NoteBook();
20Computer ct = new Tablet();
21short myfreq = 30;
```

Statica
oc è Object - cerco in Object setCPU(short) o la firma piu simile - in Object non trovo nulla - errore
Dinamica

Statica
cc è Computer - cerco in COmputer setCPU(short) o la firma piu simile - trovo setCPU(int) che è la piu simile - la prendo
Dinamica
cc è Computer - cerco la firma setCPU(int) in computer - stampo C

Statica
cn è Computer - cerco in COmputer setCPU(short) o la firma piu simile - trovo setCPU(int) che è la piu simile - la prendo
Dinamica
cn è NoteBook - cerco in NoteBook la firma presa in mod. Statica - La trovo all'interno di NoteBook - Stampo N

Statica
ct è Computer - cerco in COmputer setCPU(short) o la firma piu simile - trovo setCPU(int) che è la piu simile - la prendo
Dinamica
ct è Tablet - cerco in Tablet la firma presa in mod. Statica - Non la trovo (non va bene che il parametro sia short, io cerco INT) - risalgo alla classe superiore (ovvero Computer) - trovo la firma presa in mod. Statica - stampo C

Quale è lpuput prodotto dalle sequenti istruzioni (errore se pensi ci sia un errore)

oc.setCPU(myfreq) errore *(2 Punti)*
cc.setCPU(myfreq) C *(1 Punto)*
cn.setCPU(myfreq) N *(1 Punto)*
ct.setCPU(myfreq) C *(4 Punti)*

2. overriding di equals *(8 Punti)*

Data la seguente classe e gli oggetti definiti come segue

```
1 public class A {
2   String name;
3
4   public A(String s) {
5     name = s;
6   }
7
8   public boolean equals(A a) {
9     return this.name == a.name;
10  }
11}
12String p1 = new String("pippo");
13String p2 = new String("pippo");
14Object o = new A(p1);
15A a1 = new A(p1);
16A a2 = new A(p2);
```

1- p1 è String - cerco in String il metodo equals(Object) che è presente - prima si verifica se hanno lo stesso riferimento - non è cosi - allora si verifica che Object sia un istanza di String - Non è cosi (essendo un istanza di A) - FALSE (solo successivamente si sarebbe verificato il CONTENUTO)
2- o è Object - cerco in Object il metodo equals(Object) che è presente. Viene verificato se i due oggetti hanno lo stesso riferimento- è così (dato che sono lo stesso oggetto)- TRUE
3- o è Object - cerco in Object il metodo equals(Object) (NB:è il metodo piu simile a equals(Integer)) che è presente. I due oggetti non hanno lo stesso riferimento - FALSE
4- o è Object - cerco in Object il metodo equals(Object) (NB:è il metodo piu simile a equals(A)) che è presente. I due oggetti non hanno lo stesso riferimento - FALSE

Quale è l'output prodotto dalle seguenti istruzioni (metti errore se pensi ci sia un errore)?

```
System.out.println(p1.equals(o));false (1 Punto)
 System.out.println(o.equals(o));true (1 Punto)
 System.out.println(o.equals(new Integer(5)));false (1 Punto)
```

```
System.out.println(o.equals(a1));false (1 Punto)
System.out.println(a1.equals(a1));true (1 Punto)
System.out.println(a1.equals(a2));false (3 Punti)
```

5- a1 è A. Cerco in A equals(A). E' presente. Notiamo come venga confrontato il campo name dei due oggetti. Questo campo è un oggetto String. Facendo == per confrontarli si sta semplicemente verificando se i due oggetti String hanno lo stesso riferimento - E' così dato che sono parametri di uno stesso oggetto (sempre a1) - TRUE
6- a1 e a2 sono A. Cerco in A equals(A). E' presente. Notiamo come venga confrontato il campo name dei due oggetti. Questo campo è un oggetto String. Facendo == per confrontarli si sta semplicemente verificando se i due oggetti String hanno lo stesso riferimento - Non sarà così (nonostante il CONTENUTO dei parametri stringa dei due oggetti a1 e a2 sia uguale) perchè p1 e p2 sono due oggetti stringa distinti - FALSE

3. C++ virtual functions ed ereditarietà *(14 Punti)*

Date le seguenti classi

```
1 class A{
2 private:
3         void pri(){cout << "A" << endl;}
4 public:
5         virtual void pub(){ cout << "A" << endl;}
6 };
7
8 class APRI1: private A{
9 public:
10        void pri(){cout << "APRI1" << endl;}
11
12};
13
14class APRI2: private A{
15public:
16        virtual void pub(){ cout << "APRI2" << endl;}
17};
18
19class APUB1: public A{
20public:
21        void pri(){cout << "APUB1" << endl;}
22};
23
24class APUB2: public A{
25public:
26        virtual void pub(){ cout << "APUB2" << endl;}
27};
```

Scrivi lì'ouptput delle seguenti coppie di istruzioni. Se pensi ci sia un errore scrivi errore e ignora l'istruzione (solo quella che dà errore).

A a1; a1.pri();errore *(1 Punto)*
APRI1 a2; a2.pri();APRI1 *(1 Punto)*
APRI2 a3; a3.pri();errore *(1 Punto)*
APUB1 a4; a4.pri();APUB1 *(1 Punto)*
APUB2 a5; a5.pri();errore *(1 Punto)*
a1 = a2; a1.pub();errore *(1 Punto)*
a1 = a3; a1.pub();errore *(1 Punto)*
a1 = a4; a1.pub();A *(1 Punto)*
a1 = a5; a1.pub();A *(1 Punto)*
A* p = &a1; p -> pub();A *(1 Punto)*
p = &a2;p -> pub();errore *(1 Punto)*
p = &a3;p -> pub();errore *(1 Punto)*
p = &a4;p -> pub();A *(1 Punto)*
p = &a5;p -> pub();APUB2 *(1 Punto)*

con puntatore viene fatto binding dinamico
sena puntatore viene sempre richiamata pub di A
(QUI ho slicing: i sottotipi vengono tagliati e considerati solo come A)
e se la classe estende A in modo private -> errore
altrimenti ->stampa A

Forse in statico cerca metodo per A in Apri2 ma questo è private quindi errore e non arriva a fare binding dinamico accorgendosi che p in realta è apri2 e che ha un suo metodo public

4. Passaggio per riferimento in c++ *(9 Punti)*

Data la seguente funzione

```
1void foo(int& x, int& y) {
2        x = y;
3        x++;
4        y--;
5}
```

Qual'è l'output prodotto dalle seguenti istruzioni? Se pensi conteng un errore, scrivi errore

int main() {

```
int a = 10;
int b = 30;
int& h = b;
foo(a,b);
cout << a << endl;31 (1 Punto)
cout << b << endl;29 (1 Punto)
cout << h << endl;29 (1 Punto)
foo(h,b);
cout << a << endl;31 (1 Punto)
cout << b << endl;29 (1 Punto)

cout << h << endl;29 (1 Punto)

h = 40;

cout << a << endl;31 (1 Punto)
cout << b << endl;50 (1 Punto)

cout << h << endl;50 (1 Punto)

return 0;
}
```

5. Overriding/Overloading *(4 Punti)*

Dato il seguente codice

```
1 class Value{}
2 class SmallValue extends Value{}
3
4 class Elaboratore{
5   Value getVal(){return new Value();}
6 }
7
8 class Phone extends Elaboratore{
9   SmallValue getVal(){return new SmallValue();}
10}
```

si puo ridefinire il ritorno di una funzione con il suo sottotipo e sto comunque facendo overiding

nb:overloading riscrivo il metodo aggiungendo paramtri, anche nella stessa classe

Quali di queste sono giuste

☐ **Phone fa overloading del metodo getVal di Elaboratore**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
☐ **Phone contiene un errore: non può definire getVal in questo modo!**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
☑ **Phone è una sottoclasse di Elaboratore**	***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***
☑ **Phone fa overriding del metodo getVal di Elaboratore**	***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***

6. Overriding/Overloading (2) *(3 Punti)*

Dato il seguente codice

```
1class Elaboratore{
2  void setQuantity(int q){}
3}
4
5class Phone extends Elaboratore{
6  void setQuantity(long l){}
7}
```

non aggiungo parametri etc
MA CAMBIO IL TIPO DEL PARAMETRO
OVERLOADING!!!

Quali di queste sono giuste

☐ **Phone contiene un errore: non può definire setQuantity in questo modo!**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***

☐ **Phone fa overloading del metodo setQuantity di Elaboratore**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
☑ **Phone fa overriding del metodo setQuantity di Elaboratore**	***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***

7. Passaggio di array in C *(7 Punti)*

Data la seguente funzione

```
1 void f(int a[]){
2         printf("%d\n",sizeof(a));
3         a = a +1;
4 }
5
6
7 int main(void) {
8         int p[] = {10,20,30};
9         printf("%d\n",sizeof(p));
10        f(p);
11        printf("%d\n",*p);
12        return EXIT_SUCCESS;
13}
```

Qual'è l'output prodotto dalle seguenti istruzioni (in ordine di esecuzione)? Se pensi contenga un errore, scrivi errore

Assumi che un puntatore vale 4 byte come anche un intero (32 bit).

nel main:

printf("%d\n",sizeof(p));12 *(1 Punto)*

in f:

printf("%d\n",sizeof(a))4 *(4 Punti)*

nel main di nuovo

printf("%d\n",*p);10 *(2 Punti)*

Il puntatore viene passato per valore (non per riferimento)
se faccio a[0]=2 nella funzione allora modifico in memoria e una volta tornato nel main a[0] è uguale a 2. Ma se faccio a=a+1 mi sposto in avanti di una cella solo all'interno della funzione. Tornato nel main il puntatore punterà ancora al primo elemnto dell'array.

facendo *p (deferenziazzione) ottengo il primo elemento salvato nell'array

Anteprima di test

parte teorica

Data: Fri Jul 4 13:22:51 2014 Punteggi massimi: 53

1. Dynamic Binding Java *(8 Punti)*

Date le segenti dichiarazioni:

```
1 class Computer {
2   void setCPU(int l) {
3     System.out.println("C");
4   }
5 }
6 class NoteBook extends Computer {
7   void setCPU(int l) {
8     System.out.println("N");
9   }
10}
11class Tablet extends Computer {
12  void setCPU(short l) {
13    System.out.println("T");
14  }
15}
16...
17Object oc = new Computer();
18Computer cc = new Computer();
19Computer cn = new NoteBook();
20Computer ct = new Tablet();
21short myfreq = 30;
```

Quale è lpuput prodotto dalle sequenti istruzioni (errore se pensi ci sia un errore)

oc.setCPU(myfreq) errore *(2 Punti)* errore perche cerca in object setCPU che però non esiste

cc.setCPU(myfreq) C *(1 Punto)*

cn.setCPU(myfreq) N *(1 Punto)*

ct.setCPU(myfreq) C *(4 Punti)* statico cerca in Computer e trova il metodo con int
dinamico cerca in Tablet la firma con int, non la trova e allora sale nonstante myfreq sia short

2. overriding di equals *(8 Punti)*

Data la seguente classe e gli oggetti definiti come segue

```
1 public class A {
2   String name;
3
4   public A(String s) {
5     name = s;
6   }
7
8   public boolean equals(A a) {
9     return this.name == a.name;
10  }
11}
12String p1 = new String("pippo");
13String p2 = new String("pippo");
14Object o = new A(p1);
15A a1 = new A(p1);
```

questo == è sbagliato per le stringhe. Confronta il riferiemtno e non il contenuto. Bisognerebbe usare compareTo

string ha equals che come prima cosa guarda se l'oggetto mandato è instaceof String, se non lo è subito false

object invece confronta solo se hanno lo stesso riferimento (quindi solo se sono lo stesso oggetto)

```
16A a2 = new A(p2);
```

Quale è l'output prodotto dalle seguenti istruzioni (metti errore se pensi ci sia un errore)?

```
System.out.println(p1.equals(o));false (1 Punto)
 System.out.println(o.equals(o));true (1 Punto)
 System.out.println(o.equals(new Integer(5)));false (1 Punto)
 System.out.println(o.equals(a1));false (1 Punto)
 System.out.println(a1.equals(a1));true (1 Punto)
 System.out.println(a1.equals(a2));false (3 Punti)
```

stesso riferimento quindi true

stesso riferimento quindi true

3. C++ virtual functions ed ereditarietà *(14 Punti)*

Date le seguenti classi

```
1 class A{
2 private:
3         void pri(){cout << "A" << endl;}
4 public:
5         virtual void pub(){ cout << "A" << endl;}
6 };
7
8 class APRI1: private A{
9 public:
10        void pri(){cout << "APRI1" << endl;}
11
12};
13
14class APRI2: private A{
15public:
16        virtual void pub(){ cout << "APRI2" << endl;}
17};
18
19class APUB1: public A{
20public:
21        void pri(){cout << "APUB1" << endl;}
22};
23
24class APUB2: public A{
25public:
26        virtual void pub(){ cout << "APUB2" << endl;}
27};
```

Scrivi lì'ouptput delle seguenti coppie di istruzioni. Se pensi ci sia un errore scrivi **errore** e ignora l'istruzione (solo quella che dà errore).

A a1; a1.pri();errore *(1 Punto)* errore: è privata

APRI1 a2; a2.pri();APRI1 *(1 Punto)* questa è pubblica, eredita da A i suoi metodi (tranne quelli private) che diventano private

APRI2 a3; a3.pri();errore *(1 Punto)* APRI3 erdita i metodi di A che diventano private quindi errore

APUB1 a4; a4.pri();APUB1 *(1 Punto)*

APUB2 a5; a5.pri();errore *(1 Punto)*

mentalmente: prima fai ereditare a APUB1 i metodi di A (non i private) e nel caso falli diventare private se questa eredita in modo private
poi fai slicign e quindi elimina tutto cio che non è stato ereditato da A
Ti rimane il metodo pub di A che è diventato private

a1 = a2; a1.pub();errore *(1 Punto)*

a1 = a3; a1.pub();errore *(1 Punto)*

a1 = a4; a1.pub();A *(1 Punto)*

a1 = a5; a1.pub();A *(1 Punto)*

A* p = &a1; p -> pub();A *(1 Punto)*

p = &a2;p -> pub();errore *(1 Punto)*

p = &a3;p -> pub();errore *(1 Punto)* binding statico: cerca pub di A e trovo che nella classe ereditata è private subito errore. Non arriva a capire che ce anche quella public di APRI2

p = &a4;p -> pub();A *(1 Punto)*

p = &a5;p -> pub();APUB2 *(1 Punto)* qui avviene bindinf dinamico

4. Passaggio per riferimento in c++ *(9 Punti)*

Data la seguente funzione

```
1void copy(int& x, int& y) {
2        x = y;
3        x--;
4        y++;
5}
```

Qual'è l'output prodotto dalle seguenti istruzioni? Se pensi conteng un errore, scrivi errore

int main() {
int a = 5;
int b = 6;
int& h = a;

copy(a,b);
cout << a << endl;5 *(1 Punto)*
cout << b << endl;7 *(1 Punto)*
cout << h << endl;5 *(1 Punto)*
copy(h,b);
cout << a << endl;6 *(1 Punto)*
cout << b << endl;8 *(1 Punto)*

cout << h << endl;6 *(1 Punto)*

b=7;

cout << a << endl;6 *(1 Punto)*
cout << b << endl;7 *(1 Punto)*

cout << h << endl;6 *(1 Punto)*

return 0;
}

5. Overriding/Overloading *(4 Punti)*

Dato il seguente codice

```
1 class Value{}
2 class SmallValue extends Value{}
3
4 class Elaboratore{
5   Value getVal(){return new Value();}
6 }
7
8 class Phone extends Elaboratore{
9   SmallValue getVal(){return new SmallValue();}
10}
```

Quali di queste sono giuste

☐ Phone fa overloading del metodo getVal di Elaboratore	*(Selezionato = 0 Punti, Non selezionato = 1 Punto)*
☐ Phone contiene un errore: non può definire getVal in questo modo!	*(Selezionato = 0 Punti, Non selezionato = 1 Punto)*
☑ Phone è una sottoclasse di Elaboratore	*(Selezionato = 1 Punto, Non selezionato = 0 Punti)*
☑ Phone fa overriding del metodo getVal di Elaboratore	*(Selezionato = 1 Punto, Non selezionato = 0 Punti)*

6. Overriding/Overloading (2) *(3 Punti)*

Dato il seguente codice

```
1class Elaboratore{
2  void setQuantity(int q){}
3}
4
5class Phone extends Elaboratore{
6  void setQuantity(long l){}
7}
```

Quali di queste sono giuste

☐ Phone contiene un errore: non può definire setQuantity in questo modo!	*(Selezionato = 0 Punti, Non selezionato = 1 Punto)*
☑ Phone fa overloading del metodo setQuantity di Elaboratore	*(Selezionato = 1 Punto, Non selezionato = 0 Punti)*
☐ Phone fa overriding del metodo setQuantity di Elaboratore	*(Selezionato = 0 Punti, Non selezionato = 1 Punto)*

7. Passaggio di array in C *(7 Punti)*

Data la seguente funzione

```
1 void f(int a[]){
2         printf("%d\n",sizeof(a));
3         a = a +1;
4 }
5
6
7 int main(void) {
8         int p[] = {10,20,30};
9         printf("%d\n",sizeof(p));
10        f(p);
11        printf("%d\n",*p);
12        return EXIT_SUCCESS;
13}
```

Qual'è l'output prodotto dalle seguenti istruzioni (in ordine di esecuzione)? Se pensi contenga un errore, scrivi errore

Assumi che un puntatore vale 4 byte come anche un intero (32 bit).

nel main:

printf("%d\n",sizeof(p));12 *(1 Punto)*

in f:

printf("%d\n",sizeof(a))4 *(4 Punti)*

nel main di nuovo

printf("%d\n",*p);10 *(2 Punti)*

Anteprima di test

test info3 21.11.16

Data: Mon Jan 9 16:12:07 2017 Punteggi massimi: 61

1. Dynamic Binding Java (2) *(15 Punti)*

Date le segenti dichiarazioni:

```
1 class Persona {
2   void setAge(int l) {
3     System.out.println("P");
4   }
5 }
6
7 class Studente extends Persona {
8   void setAge(int l) {
9     System.out.println("S");
10  }
11}
12
13class Anziano extends Persona {
14  void setAge(long l) {
15    System.out.println("A");
16  }
17}
18
19...
20Object op = new Persona();
21Persona pp = new Persona();
22Persona ps = new Studente();
23Persona pa = new Anziano();
24Anziano aa = new Anziano();
25int age = 30;
26long ageL = 200;
```

NB: lo static binding puo SOLO SALIRE una classe
quindi se da una classe piu bassa non trova il metodo adatto al 100%
puo provare a salire
una volta arrivati alla classe madre, se non ne trova uno al 100% allora si adatta (es. invio uno short e il metodo è int
scelgo comunque questo perchè non ho alternative)

Qual è l'input prodotto dalle sequenti istruzioni (errore se pensi ci sia un errore)?

op.setAge(age);errore *(2 Punti)*

pp.setAge(age);P *(1 Punto)*
ps.setAge(age);S *(1 Punto)*
pa.setAge(age);P *(4 Punti)*

errore perchè long è piu grande di int, non è compatibile
ok se da uno piu piccolo (es short) provo a mettere in uno piu grande

pa.setAge(ageL);errore *(1 Punto)*
aa.setAge(age);P *(4 Punti)* BINDING puo solo SALIRE una classe!!!
aa.setAge(ageL);A *(2 Punti)*

2. overriding di equals (4) *(9 Punti)*

Data la seguente classe e gli oggetti definiti come segue

```
1 public class Complex {
2   int re;
3   int im;
4
5   public Complex (int re, int im) {
6     this.re = re;
7     this.im = im;
8   }
9
10  public boolean equals(Complex a) {
11    return this.re == a.re && this.im == a.im;
12  }
13}
14...
15Object o = new Complex (1,2);
16Complex p1 = new Complex (1,2);
17Complex p2 = new Complex (2,1);
18Complex p3 = new Complex (2,1);
```

Quale è l'output prodotto dalle seguenti istruzioni (metti errore se pensi ci sia un errore)?

`System.out.println(o.equals(p1));false` ***(2 Punti)***
`System.out.println(p1.equals(o));false` ***(3 Punti)*** p1 non ha un equals chw accetta Object, allora salgo la classe alla ricerca di questa firma
`System.out.println(o.equals(p2));false` ***(2 Punti)*** la trovo in object che confronta solo il riferiemnto
`System.out.println(p2.equals(p3));true` ***(1 Punto)***
`System.out.println(p2.equals(p2));true` ***(1 Punto)***

3. passaggio parametri *(1 Punto)*

int foo(int x) { ... }

la variabile x viene passata per

[] ◉ valore *(1 Punto)*
[] ○ riferimento *(0 Punti)*

4. Overriding/Overloading (1) *(4 Punti)*

Dato il seguente codice

```
1 class Value {}
2 class SmallValue extends Value {}
3
4 class Computer {
5     Value getVal() {
6         return new Value();
7     }
8 }
9
10class NoteBook extends Computer {
11    SmallValue getVal() {
12        return new SmallValue();
13    }
14}
```

Quali di queste sono giuste

☑ **Notebook fa overriding del metodo getVal di Computer**	***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***
☐ **Notebook fa overloading del metodo getVal di Computer**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
☑ **NoteBook è una sottoclasse di Computer**	***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***
☐ **Notebook contiene un errore: non può definire getVal in questo modo!**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***

5. Ridefinizione di metodi con classi (3) MQ *(3 Punti)*

```
1 class Sportivo{
2    public Calciatore m(){return null;}
3 }
4 class Calciatore extends Sportivo {
5    public Calciatore m(){return null;}
6 }
7 class Pallavolista extends Sportivo {
8    private Calciatore m(){return null;}
9 }
10class Golfista extends Sportivo {
11   public Sportivo m(){return null;}
12}
```

la visibilita in java puo solo aumentare, quindi errore rendere private

sto ridefinendo il tipo di ritorno ma questo tipo di ritorno è piu grande del metodo originale! non va bene
lo posso fare con uguale tipo di ritorno o con un tipo "piu piccolo"

Quali di questi metodi sono redefiniti in modo sbagliato (errore in compilazione)?

- [x] **il metodo m di Pallavolista** ***(Selezionato = 1 Punto, Non selezionato = -1 Punto)***
- [x] **il metodo m di Golfista** ***(Selezionato = 1 Punto, Non selezionato = -1 Punto)***
- [ ] **il metodo m di Calciatore** ***(Selezionato = -1 Punto, Non selezionato = 1 Punto)***

6. C++ virtual functions ed ereditarietà (2) *(14 Punti)*

Date le seguenti classi

```
1 class Z{
2 public:
3         virtual void m(){ cout << "Z" << endl;}
4 };
5
6 class ZPRI1: private Z{
7 };
8
9 class ZPRI2: private Z{
10public:
11        virtual void m(){ cout << "ZPRI2" << endl;}
12};
13
14class ZPUB1: public Z{
15};
16
17class ZPUB2: public Z{
18public:
19        virtual void m(){ cout << "ZPUB2" << endl;}
20};
```

Scrivi l'ouptput delle seguenti coppie di istruzioni. ERR se pensi ci sia un errore.

Z a1; a1.m();Z *(1 Punto)*
ZPRI1 a2; a2.m();ERR *(1 Punto)*
ZPRI2 a3; a3.m();ZPRI2 *(1 Punto)* a3 è ZPRI2, nonstante erediti il metodo m di Z che diventa private questa ha un suo metodo m public. Quindi no errore
ZPUB1 a4; a4.m();Z *(1 Punto)*
ZPUB2 a5; a5.m();ZPUB2 *(1 Punto)*
a1 = a2; a1.m();ERR *(1 Punto)*
a1 = a3; a1.m();ERR *(1 Punto)* errore perche nella classe ereditata ho public m() di ZPRI2 e private m() di Z, elimino (slicing) tutto cio che appartiene a ZPRI2 e mi rimane private m()
a1 = a4; a1.m();Z *(1 Punto)*

a1 = a5; a1.m();Z *(1 Punto)*
Z* p = &a1; p -> m();Z *(1 Punto)*
p = &a2;p -> m();ERR *(1 Punto)*
p = &a3;p -> m();ERR *(1 Punto)* errore perche binding statico cerca il metodo ereditato (quello di Z e non ZPRI2) che è diventato private
p = &a4;p -> m();Z *(1 Punto)*
p = &a5;p -> m();ZPUB2 *(1 Punto)*

7. C++ virtual functions ed ereditarietà - calls (2) *(15 Punti)*

Date le seguenti classi e le funzioni definite sotto.

```
1 #include <iostream>
2 using namespace std;
3 class veicolo {
4 private:
5         int pri() {        return 1;        }
6 public:
7         int pub() {        return 2;        }
8
9         virtual int vpub() {        return 3;        }
10};
11
12class camper: private veicolo {
13public:
14        int vpub() {         return 5;        }
15};
16
17class automobile: public veicolo {
18private:
19        int pri() {                return 6;        }
20public:
21        int vpub() {                return 7;        }
22};
23void f_camper(camper c) {
24        cout << c.pub();
25        cout << c.pri();
26        cout << c.vpub() << endl;
27}
28void f_p_camper(camper* c) {
29        cout << c->pub();
30        cout << c->pri();
31        cout << c->vpub() << endl;
32}
33void f_r_camper(camper& c) {
34        cout << c.pub();
35        cout << c.pri();
36        cout << c.vpub() << endl;
```

```
37}
```

Scrivi l'ouptput delle seguenti istruzioni. Se una funzione chiamata f_* contiene un errore, ignora solo la riga della f_* che contiene l'errore. Se una delle seguenti istruzioni è sbagliata (anche se f_* chiamata fosse corretta), scrivi ERR.

```
int main() {
veicolo v;
camper c;
automobile a;
```

sarebbe stato ok (e avrei fatto slicing) se avessi avuto f_(Veicolo v)

f_camper(v);ERR *(1 Punto)* errore: veicolo è piu grande di camper

f_camper(c);5 *(3 Punti)* i pirmi due comandi danno errore, l'ultimo restituisci 5

f_camper(a);ERR *(1 Punto)* errore: automobile e camper sono sullo stesso livello

//

f_p_camper(&v);ERR *(1 Punto)* classe camper, elimino tutto cio che non e di Veicolo, ho che è diventato tutto private->errore

f_p_camper(&c);5 *(3 Punti)* i pirmi due comandi danno errore, l'ultimo restituisci 5

f_p_camper(&a);ERR *(1 Punto)* errore, non mi rimane nessun metodo se faccio slicign ed elimino da capmer tutto quello che non è di Automobile

//

f_r_camper(v);ERR *(3 Punti)*

f_r_camper(c);5 *(1 Punto)* questo a finaco è equivalente alla scrittura sopra

f_r_camper(a);ERR *(1 Punto)*

}

Anteprima di test

parte teorica aprile 16

Data: Mon Jan 9 16:13:47 2017 Punteggi massimi: 34

1. overriding di equals (1) *(5 Punti)*

Data la seguente classe e gli oggetti definiti come segue

```
1 public class A {
2   String name;
3
4   public A(String s) {
5     name = s;
6   }
7
8   public boolean equals(A a) {
9     return this.name.equals(a.name);
10  }
11}
12
13String pippo = "pippo";
14Object o = new A(pippo);
15A a1 = new A("pippo");
16A a2 = new A(pippo);
```

Quanto valgono (metti errore se pensi ci sia un errore)?

pippo.equals(o) false *(1 Punto)*
o.equals(o) true *(1 Punto)*
o.equals(a1) false *(1 Punto)*
o.equals(a2) false *(1 Punto)*
a1.equals(a2) true *(1 Punto)*

2. Dynamic Binding Java (1) *(4 Punti)*

Date le segenti dichiarazioni:

```
1 class Elaboratore {
2   void setCPU(int l) {
3     System.out.println("E");
4   }
```

```
5 }
6
7 class Phone extends Elaboratore {
8   void setCPU(int l) {
9     System.out.println("P");
10  }
11}
12
13class Computer extends Elaboratore {
14  void setCPU(short l) {
15    System.out.println("C");
16  }
17}
18
19...
20Object oe = new Elaboratore ();
21Elaboratore ee = new Elaboratore ();
22Elaboratore ep = new Phone ();
23Elaboratore ec = new Computer ();
24short myfreq = 30;
```

Quale è l' input prodotto dalle sequenti istruzioni (errore se pensi ci sia un errore)?

oe.setCPU(myfreq) errore *(1 Punto)*
ee.setCPU(myfreq) E *(1 Punto)*
ep.setCPU(myfreq) P *(1 Punto)*
ec.setCPU(myfreq) E *(1 Punto)*

3. passaggio parametri *(1 Punto)*

int foo(int x) { ... }

la variabile x viene passata per

[] ◉ valore *(1 Punto)*
[] ○ riferimento *(0 Punti)*

4. Return result address *(1 Punto)*

Che cos'è il return result-address?

[] ◉ un campo contenente l'indirizzo dove salvare il risultato della funzione *(1 Punto)*

[] ○ **un campo contenente l'indirizzo della funzione chiamata "return"** ***(-1 Punti)***

[] ○ **un campo contenente l'indirizzo della prima istruzione da eseguire quando la funzione termina** ***(-1 Punti)***

[] ○ **un campo contenente l'istruzione return della funzione** ***(-1 Punti)***

5. Overriding/Overloading (0) *(4 Punti)*

Dato il seguente codice

```
1 class Value {}
2 class SmallValue extends Value {}
3
4 class Elaboratore {
5     Value getVal() {
6         return new Value();
7     }
8 }
9
10class Phone extends Elaboratore {
11    SmallValue getVal() {
12        return new SmallValue();
13    }
14}
```

Quali di queste affermazioni sono giuste?

☑ **Phone fa overriding del metodo getVal di Elaboratore** ***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***
☐ **Phone contiene un errore: non può definire getVal in questo modo!** ***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
☑ **Phone è una sottoclasse di Elaboratore** ***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***
☐ **Phone fa overloading del metodo getVal di Elaboratore** ***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***

6. Ridefinizione di metodi con classi (1) *(3 Punti)*

```
1 class Veicolo{
2    public Auto m(){return null;}
3 }
4 class Auto extends Veicolo {
5    public Auto m(){return null;}
6 }
7 class Bicicletta extends Veicolo {
8    private Auto m(){return null;}
```

```
9 }
10class Autobus extends Veicolo {
11   public Veicolo m(){return null;}
12}
```

Quali di questi metodi sono redefiniti in modo sbagliato (errore in compilazione)?

- ☑ **il metodo m di Autobus** ***(Selezionato = 1 Punto, Non selezionato = -1 Punto)***
- ☐ **il metodo m di Auto** ***(Selezionato = -1 Punto, Non selezionato = 1 Punto)***
- ☑ **il metodo m di Bicicletta** ***(Selezionato = 1 Punto, Non selezionato = -1 Punto)***

7. Passaggio di array in C (5) *(7 Punti)*

Data la seguente funzione

```
1 void f(int a[]){
2         printf("%d\n",sizeof(a));
3         a = a +1;
4 }
5
6
7 int main(void) {
8         int p[] = {10,20,30};
9         printf("%d\n",sizeof(p));
10        f(p);
11        printf("%d\n",*p);
12        return EXIT_SUCCESS;
13}
```

Qual'è l'output prodotto dalle seguenti istruzioni (in ordine di esecuzione)? Se pensi contenga un errore, scrivi errore

Assumi che un puntatore vale 4 byte come anche un intero (32 bit).

nel main:

printf("%d\n",sizeof(p));12 ***(1 Punto)***

in f:

printf("%d\n",sizeof(a))4 ***(4 Punti)***

nel main di nuovo

printf("%d\n",*p);10 ***(2 Punti)***

8. C++ virtual functions ed ereditarietà - calls *(9 Punti)*

Date le seguenti classi e le funzioni definite sotto.

```
1 #include <iostream>
2 using namespace std;
3 class veicolo {
4 private:
5         int pri() {        return 1;        }
6 public:
7         int pub() {        return 2;        }
8
9         virtual int vpub() {        return 3;        }
10};
11
12class camper: private veicolo {
13public:
14        int vpub() {         return 5;        }
15};
16
17class automobile: public veicolo {
18private:
19        int pri() {                return 6;        }
20public:
21        int vpub() {                return 7;        }
22};
23void f_veicolo(veicolo v) {
24        cout << v.pub();
25        cout << v.pri() ;
26      cout << v.vpub() << endl;
27}
28void f_camper(camper c) {
29        cout << c.pub();
30        cout << c.pri();
31        cout << c.vpub() << endl;
32}
33void f_automobile(automobile a) {
34        cout << a.pub();
35        cout << a.pri();
36        cout << a.vpub() << endl;
37}
38void f_p_veicolo(veicolo* v) {
39        cout << v->pub();
40        cout << v->pri();
41        cout << v->vpub() << endl;
42}
43void f_p_camper(camper* c) {
44        cout << c->pub();
45        cout << c->pri();
```

```
46        cout << c->vpub() << endl;
47}
48void f_p_automobile(automobile* a) {
49        cout << a->pub();
50        cout << a->pri();
51        cout << a->vpub() << endl;
52}
53void f_r_veicolo(veicolo& v) {
54        cout << v.pub();
55        cout << v.pri();
56        cout << v.vpub() << endl;
57}
58void f_r_camper(camper& c) {
59        cout << c.pub();
60        cout << c.pri();
61        cout << c.vpub() << endl;
62}
63void f_r_automobile(automobile& a) {
64        cout << a.pub();
65        cout << a.pri();
66        cout << a.vpub() << endl;
67}
```

Scrivi l'ouptput delle seguenti istruzioni. Se una funcione chiamata f_* contiene un errore, ignora solo la riga della f_* che contiene l'errore. Se una delle seguenti istruzioni è sbagliata (anche se f_* chiamata fosse corretta), scrivi ERR.

int main() {

veicolo v;

camper c;

automobile a;

f_veicolo(v);23 ***(1 Punto)***

f_veicolo(c);ERR ***(1 Punto)***

f_veicolo(a);23 ***(1 Punto)***

//

f_p_veicolo(&v);23 ***(1 Punto)***

f_p_veicolo(&c);ERR ***(1 Punto)***

f_p_veicolo(&a);27 ***(1 Punto)*** basta il metodo madre come virtual

//

f_r_veicolo(v);23 ***(1 Punto)***

f_r_veicolo(c);ERR ***(1 Punto)*** scrittura analoga a quella sopra

f_r_veicolo(a);27 ***(1 Punto)***

}

Anteprima di test

parte teorica 20 Gennaio 2015

Data: Mon Jan 26 12:44:27 2015 Punteggi massimi: 38

1. Overriding/Overloading (2) *(3 Punti)*

Dato il seguente codice

```
1class Elaboratore{
2  void setQuantity(int q){}
3}
4
5class Phone extends Elaboratore{
6  void setQuantity(long l){}
7}
```

Quali di queste sono giuste

☐ **Phone fa overriding del metodo setQuantity di Elaboratore**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
☐ **Phone contiene un errore: non può definire setQuantity in questo modo!**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
☑ **Phone fa overloading del metodo setQuantity di Elaboratore**	***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***

2. overriding di equals *(8 Punti)*

Data la seguente classe e gli oggetti definiti come segue

```
1 public class A {
2   String name;
3
4   public A(String s) {
5     name = s;
6   }
7
8   public boolean equals(A a) {
9     return this.name == a.name;
10  }
11}
12String p1 = new String("pippo");
13String p2 = new String("pippo");
14Object o = new A(p1);
15A a1 = new A(p1);
16A a2 = new A(p2);
```

Quale è l'output prodotto dalle seguenti istruzioni (metti errore se pensi ci sia un errore)?

```
System.out.println(p1.equals(o));false (1 Punto)
 System.out.println(o.equals(o));true (1 Punto)
 System.out.println(o.equals(new Integer(5)));false (1 Punto)
 System.out.println(o.equals(a1));false (1 Punto)
 System.out.println(a1.equals(a1));true (1 Punto)
 System.out.println(a1.equals(a2));false (3 Punti)
```

3. Passaggio per riferimento in c++ *(9 Punti)*

Data la seguente funzione

```
1void copy(int& x, int& y) {
```

```
2        x = y;
3        x--;
4        y++;
5}
```

Qual'è l'output prodotto dalle seguenti istruzioni? Se pensi conteng un errore, scrivi errore

int main() {
int a = 5;
int b = 6;
int& h = a;

copy(a,b);
cout << a << endl;5 ***(1 Punto)***
cout << b << endl;7 ***(1 Punto)***
cout << h << endl;5 ***(1 Punto)***
copy(h,b);
cout << a << endl;6 ***(1 Punto)***
cout << b << endl;8 ***(1 Punto)***

cout << h << endl;6 ***(1 Punto)***

b=7;

cout << a << endl;6 ***(1 Punto)***
cout << b << endl;7 ***(1 Punto)***

cout << h << endl;6 ***(1 Punto)***

return 0;
}

4. Dynamic Binding Java (1) *(4 Punti)*

Date le segenti dichiarazioni:

```
1 class Elaboratore {
2   void setCPU(int l) {
3     System.out.println("E");
4   }
5 }
6
7 class Phone extends Elaboratore {
8   void setCPU(int l) {
9     System.out.println("P");
10  }
11}
12
13class Computer extends Elaboratore {
14  void setCPU(short l) {
15    System.out.println("C");
16  }
17}
18
19...
20Object oe = new Elaboratore ();
21Elaboratore ee = new Elaboratore ();
22Elaboratore ep = new Phone ();
23Elaboratore ec = new Computer ();
24short myfreq = 30;
```

Quale è l' input prodotto dalle sequenti istruzioni (errore se pensi ci sia un errore)?

oe.setCPU(myfreq) errore ***(1 Punto)***
ee.setCPU(myfreq) E ***(1 Punto)***
ep.setCPU(myfreq) P ***(1 Punto)***

ec.setCPU(myfreq) E *(1 Punto)*

5. C++ virtual functions ed ereditarietà (2) *(14 Punti)*

Date le seguenti classi

```
1 class Z{
2 public:
3         virtual void m(){ cout << "Z" << endl;}
4 };
5
6 class ZPRI1: private Z{
7 };
8
9 class ZPRI2: private Z{
10public:
11        virtual void m(){ cout << "ZPRI2" << endl;}
12};
13
14class ZPUB1: public Z{
15};
16
17class ZPUB2: public Z{
18public:
19        virtual void m(){ cout << "ZPUB2" << endl;}
20};
```

Scrivi l'ouptput delle seguenti coppie di istruzioni. ERR se pensi ci sia un errore.

Z a1; a1.m();Z *(1 Punto)*
ZPRI1 a2; a2.m();ERR *(1 Punto)*
ZPRI2 a3; a3.m();ZPRI2 *(1 Punto)*
ZPUB1 a4; a4.m();Z *(1 Punto)*
ZPUB2 a5; a5.m();ZPUB2 *(1 Punto)*
a1 = a2; a1.m();ERR *(1 Punto)*
a1 = a3; a1.m();ERR *(1 Punto)*
a1 = a4; a1.m();Z *(1 Punto)*
a1 = a5; a1.m();Z *(1 Punto)*
Z* p = &a1; p -> m();Z *(1 Punto)*
p = &a2;p -> m();ERR *(1 Punto)*
p = &a3;p -> m();ERR *(1 Punto)*
p = &a4;p -> m();Z *(1 Punto)*
p = &a5;p -> m();ZPUB2 *(1 Punto)*

Anteprima di test

parte teorica 4 Febbraio 15

Data: Thu Feb 19 11:31:51 2015 Punteggi massimi: 42

1. Dynamic Binding Java *(8 Punti)*

Date le segenti dichiarazioni:

```
1 class Computer {
2   void setCPU(int l) {
3     System.out.println("C");
4   }
5 }
6 class NoteBook extends Computer {
7   void setCPU(int l) {
8     System.out.println("N");
9   }
10}
11class Tablet extends Computer {
12  void setCPU(short l) {
13    System.out.println("T");
14  }
15}
16...
17Object oc = new Computer();
18Computer cc = new Computer();
19Computer cn = new NoteBook();
20Computer ct = new Tablet();
21short myfreq = 30;
```

Qual è l'input prodotto dalle sequenti istruzioni (errore se pensi ci sia un errore)?

oc.setCPU(myfreq) errore *(2 Punti)*
cc.setCPU(myfreq) C *(1 Punto)*
cn.setCPU(myfreq) N *(1 Punto)*
ct.setCPU(myfreq) C *(4 Punti)*

2. Passaggio di array in C *(7 Punti)*

Data la seguente funzione

```
1 void f(int a[]){
2         printf("%d\n",sizeof(a));
3         a = a +1;
4 }
5
6
7 int main(void) {
8         int p[] = {10,20,30};
9         printf("%d\n",sizeof(p));
10        f(p);
11        printf("%d\n",*p);
12        return EXIT_SUCCESS;
13}
```

Qual'è l'output prodotto dalle seguenti istruzioni (in ordine di esecuzione)? Se pensi contenga un errore, scrivi errore

Assumi che un puntatore vale 4 byte come anche un intero (32 bit).

nel main:

printf("%d\n",sizeof(p));12 *(1 Punto)*

in f:

printf("%d\n",sizeof(a))4 *(4 Punti)*

nel main di nuovo

printf("%d\n",*p);10 *(2 Punti)*

3. C++ virtual functions ed ereditarietà (3) *(14 Punti)*

Date le seguenti classi

```
1 class A{
2 private:
3         void pri(){cout << "A" << endl;}
4 public:
5         virtual void pub(){ cout << "A" << endl;}
6 };
```

```
7
8 class APRI1: private A{
9 public:
10        void pri(){cout << "APRI1" << endl;}
11
12};
13
14class APRI2: private A{
15public:
16        virtual void pub(){ cout << "APRI2" << endl;}
17};
18
19class APUB1: public A{
20public:
21        void pri(){cout << "APUB1" << endl;}
22};
23
24class APUB2: public A{
25public:
26        virtual void pub(){ cout << "APUB2" << endl;}
27};
```

Scrivi l'ouptput delle seguenti coppie di istruzioni. Se pensi ci sia un errore scrivi errore e ignora l'istruzione (solo quella che dà errore).

A a1; a1.pri();errore *(1 Punto)*
APRI1 a2; a2.pri();APRI1 *(1 Punto)*
APRI2 a3; a3.pri();errore *(1 Punto)*
APUB1 a4; a4.pri();APUB1 *(1 Punto)*
APUB2 a5; a5.pri();errore *(1 Punto)*
a1 = a2; a1.pub();errore *(1 Punto)*
a1 = a3; a1.pub();errore *(1 Punto)*
a1 = a4; a1.pub();A *(1 Punto)* se avessi avuto a1.pri() => errore !
a1 = a5; a1.pub();A *(1 Punto)*
A* p = &a1; p -> pub();A *(1 Punto)*
p = &a2;p -> pub();errore *(1 Punto)*
p = &a3;p -> pub();errore *(1 Punto)*
p = &a4;p -> pub();A *(1 Punto)*
p = &a5;p -> pub();APUB2 *(1 Punto)*

4. Overriding/Overloading *(4 Punti)*

Dato il seguente codice

```
1 class Value {}
2 class SmallValue extends Value {}
3
4 class Elaboratore {
5     Value getVal() {
6         return new Value();
7     }
8 }
9
10class Phone extends Elaboratore {
11    SmallValue getVal() {
12        return new SmallValue();
13    }
14}
```

Quali di queste affermazioni sono giuste?

☑ **Phone fa overriding del metodo getVal di Elaboratore**	***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***
☑ **Phone è una sottoclasse di Elaboratore**	***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***
☐ **Phone contiene un errore: non può definire getVal in questo modo!**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
☐ **Phone fa overloading del metodo getVal di Elaboratore**	***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***

5. overriding di equals (1) *(5 Punti)*

Data la seguente classe e gli oggetti definiti come segue

```
1 public class A {
2   String name;
3
4   public A(String s) {
5     name = s;
6   }
7
8   public boolean equals(A a) {
```

```
9     return this.name.equals(a.name);
10  }
11}
12
13String pippo = "pippo";
14Object o = new A(pippo);
15A a1 = new A("pippo");
16A a2 = new A(pippo);
```

Quanto valgono (metti errore se pensi ci sia un errore)?

pippo.equals(o) false *(1 Punto)*
o.equals(o) true *(1 Punto)*
o.equals(a1) false *(1 Punto)*
o.equals(a2) false *(1 Punto)*
a1.equals(a2) true *(1 Punto)*

6. overriding di equals *(4 Punti)*

Data la seguente classe e gli oggetti definiti come segue

```
1 public class A {
2   int x;
3
4   public A(int s) {
5     x = s;
6   }
7
8   public boolean equals(A a) {
9     return this.x == a.x;
10  }
11}
12
13Object o = new A(1);
14A a1 = new A(1);
15A a2 = new A(1);
```

Quale è l'output prodotto dalle seguenti istruzioni (metti errore se pensi ci sia un errore)?

System.out.println(o.equals(a1));false *(1 Punto)*
System.out.println(a1.equals(o));false *(1 Punto)*
System.out.println(a1.equals(a1));true *(1 Punto)*
System.out.println(a1.equals(a2));true *(1 Punto)*

Anteprima di test

parte teorica 19.01.17

Data: Thu Jan 19 15:06:03 2017 Punteggi massimi: 73

1. Dynamic Binding Java (7) *(12 Punti)*

Date le segenti dichiarazioni:

```
1 class Persona{}
2 class Studente extends Persona{}
3
4 class Scuola{
5   void iscrivi(Persona l) {
6     System.out.println("S");
7   }
8 }
9
10class Liceo extends Scuola{
11  void iscrivi(Persona l) {
12    System.out.println("L");
13  }
14}
15
16class University extends Scuola{
17  void iscrivi(Studente l) {
18    System.out.println("U");
19  }
20}
21
22...
23Persona p = new Studente();
24Studente s = new Studente();
25Scuola ss = new Scuola();
26Scuola sl = new Liceo();
27Scuola su = new University();
```

Qual è l'input prodotto dalle sequenti istruzioni (errore se pensi ci sia un errore)?

ss.iscrivi(p);S *(1 Punto)*
sl.iscrivi(p);L *(1 Punto)*
su.iscrivi(p);S *(5 Punti)*

ss.iscrivi(s);S ***(1 Punto)***
sl.iscrivi(s);L ***(1 Punto)***
su.iscrivi(s);S ***(3 Punti)***

2. overriding di equals (6) MQ *(14 Punti)*

Data la seguente classe e gli oggetti definiti come segue

```
1 public class Persona{
2   String nome;
3   String cognome;
4
5   public Persona(String n, String c) {
6     this.nome = n;
7     this.cognome = c;
8   }
9
10  public boolean equals(Persona a) {
11    return this.nome.equals(a.nome) && this.cognome == a.cognome;
12  }
13}
14...
15String n = new String("Angelo");
16String c = new String("Gargantini");
17
18Object o = new Persona ("Angelo","Gargantini");
19Persona p1 = new Persona ("Angelo","Gargantini");
20Persona p2 = new Persona ("Gargantini","Angelo");
21Persona p3 = new Persona ("angelo","Gargantini");
22Persona p4 = new Persona ("Angelo","Gargantini");
23Persona p5 = new Persona (n,c);
```

Quale è l'output prodotto dalle seguenti istruzioni (metti errore se pensi ci sia un errore)?

System.out.println(o.equals(p1));false *(2 Punti)*
System.out.println(p1.equals(o));false *(3 Punti)*
System.out.println(o.equals(p2));false *(2 Punti)*
System.out.println(p2.equals(p3));false *(1 Punto)*
System.out.println(p2.equals(p2));true *(1 Punto)*

System.out.println(p1.equals(p4));true *(3 Punti)*

System.out.println(p1.equals(p5));false *(1 Punto)*

`System.out.println(p1.equals(new Persona(n,c)));false` ***(1 Punto)***

3. C++ virtual destructors ed ereditarietà (1) *(7 Punti)*

Date le seguenti classi:

```
1 struct A {
2              A() { cout << "A+"; }
3     virtual ~A() { cout << "A-"; }
4 };
5
6 struct B {
7      B() { cout << "B+"; }
8     ~B() { cout << "B-"; }
9 };
10
11struct C {
12     C() { cout << "C+"; }
13    ~C() { cout << "C-"; }
14};
15
16struct X : A, B, protected C {
17     X() { cout << "X+"; }
18    ~X() { cout << "X-"; }
19};
```

costruttore viene chiamato in ordine a partire dalle classi Madri e solo infine viene chiamato il costruttore del figlio
Per il distruttore invece avviene il contrario. Si parte dal figlio e si risale alle classi superiori
NB: l'ordine per costruttore se ho piu classi madre è da sx->dx
per il distruttore l'ordine è da dx->sx

i costruttori madre vengono chiamati della classe che si trova col il new (ovvero a dx dell'uguale)

il distruttore viene chiamato solo del tipo che sta a sx dell'uguale. se questo è virtual vengono distrutti tutti

Scrivi l'output delle seguenti istruzioni (ERR se pensi che ci sia un errore):

A* a = new A; delete a;A+A- ***(1 Punto)***

A* a = new X; delete a;A+B+C+X+X-C-B-A- ***(1 Punto)***

qui vengono richiamati tutti i distruttori perche il distruttore di A(che è quello che viene richiamato) è virtual

B* b = new B; delete b;B+B- ***(1 Punto)***

B* b = new X; delete b;A+B+C+X+B- ***(1 Punto)***

C* c = new C; delete c;C+C- ***(1 Punto)***

C* c = new X; delete c;ERR ***(1 Punto)***

L'ereditarietà protected rende invisibile la relazione tra X e C all'esterno della classe X o delle sue derivate. Pertanto, non puoi usare un puntatore di tipo C* per riferirti a un'istanza di X al di fuori di X stessa.

X* x = new X; delete x;A+B+C+X+X-C-B-A- ***(1 Punto)***

costruttore X-> prima superclassi, distruttore X->prima X poi superclassi
vegnono richimati tutti i distruttori anche se non è virtual dato che è la classe piu bassa

4. Ridefinizione di metodi con classi (0) *(3 Punti)*

```
1 class Veicolo{
2    public Veicolo get(){return null;}
```

```
3 }
4 class Auto extends Veicolo {
5    public Veicolo get(){return null;}
6 }
7 class Bicicletta extends Veicolo {
8    private Veicolo get(){return null;}
9 }
10class Autobus extends Veicolo {
11   public Autobus get(){return null;}
12}
```

Quali di questi metodi sono redefiniti in modo sbagliato (errore in compilazione)?

- ☐ **il metodo get di Auto** ***(Selezionato = -1 Punto, Non selezionato = 1 Punto)***
- ☐ **il metodo get di Autobus** ***(Selezionato = -1 Punto, Non selezionato = 1 Punto)***
- ☑ **il metodo get di Bicicletta** ***(Selezionato = 1 Punto, Non selezionato = -1 Punto)***

5. Passaggio di array in C (8) MQ *(7 Punti)*

Data la seguente funzione

```
1 void f(int a[], int n) {
2         printf("%d\n", n);
3         printf("%d\n", sizeof(a));
4         a = a + 1;
5         *a = *a + 1;
6 }
7
8 int main(void) {
9         int p[] = { 10, 20, 30 };
10        f(p, sizeof(p));
11        printf("%d\n", *p );
12        return 0;
13}
```

Qual è l'output prodotto dalle seguenti istruzioni (in ordine di esecuzione)? Se pensi contenga un errore, scrivi errore

Assumi che un puntatore vale 4 byte come anche un intero (32 bit).

in f:

printf("%d\n", n);12 ***(1 Punto)***
printf("%d\n", sizeof(a));4 ***(4 Punti)***

in main:

printf("%d\n", *p);10 *(2 Punti)*

6. C++ virtual functions ed ereditarietà (6) - FG *(14 Punti)*

Date le seguenti classi

```
1 class X {
2 private:
3     void pri() { cout << "X" << endl; }
4 public:
5     virtual void pub() { cout << "X" << endl; }
6 };
7
8 class Y : private X {
9 public:
10    void pri() { cout << "Y" << endl; }
11
12};
13
14class Z : private X {
15public:
16    virtual void pub() { cout << "Z" << endl; }
17};
18
19class V : public X {
20public:
21    void pri() { cout << "V" << endl; }
22};
23
24class W : public X {
25public:
26    virtual void pub() { cout << "W" << endl; }
27};
```

Scrivi l'ouptput delle seguenti coppie di istruzioni. Se pensi ci sia un errore scrivi ERR e ignora l'istruzione (solo quella che dà errore).

X x; x.pri();ERR *(1 Punto)*
Y y; y.pri();Y *(1 Punto)*
Z z; z.pri();ERR *(1 Punto)*
V v; v.pri();V *(1 Punto)*
W w; w.pri();ERR *(1 Punto)*
x = y; x.pub();ERR *(1 Punto)*
x = z; x.pub();ERR *(1 Punto)*
x = v; x.pub();X *(1 Punto)*

x = w; x.pub();X *(1 Punto)*
X* p = &x; p->pub();X *(1 Punto)*
p = &y; p->pub();ERR *(1 Punto)*
p = &z; p->pub();ERR *(1 Punto)*
p = &v; p->pub();X *(1 Punto)*
p = &w; p->pub();W *(1 Punto)*

7. Java varargs (2) MR *(5 Punti)*

Selezionare, fra i seguenti metodi, quello/i corretto/i (cioè che NON dà/danno errore in compilazione):

- [x] **static void prova1(int i, String... s) { /**/ }** ***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***
- [ ] **static void prova2(String ...s, int i) { /**/ }** ***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
- [ ] **static void prova3(int ...i, String ... s) { /**/ }** ***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
- [ ] **static void prova4(int i,int ...j, String ...s) { /**/ }** ***(Selezionato = 0 Punti, Non selezionato = 1 Punto)***
- [x] **static void prova5(String s, int ...i){ /**/ }** ***(Selezionato = 1 Punto, Non selezionato = 0 Punti)***

posso averne uno solo e deve essere l'ultimo argomento

8. Java generics - tipi (4) MR *(11 Punti)*

```
1class Shape { /* ... */ }
2class Circle extends Shape { /* ... */ }
3class Rectangle extends Shape { /* ... */ }
```

Dire, per ognuno dei seguenti pezzi di programma, se è corretto o se presenta errori. Nel caso di errore, indicare la PRIMA linea di codice che contiene l'errore:

// -- Codice 1:

class Node<T> { /* ... */ }

Node<Circle> nc = new Node<Circle>();

Node<Shape> ns = nc; errore istruzione 3 *(1 Punto)*

non esiste covarianza con i generics

// -- Codice 2:

qui errore

class Node<?> { /* ... */ }

Node<Circle> nc = new Node<Circle>();

Node<Circle> ns = nc; errore istruzione 1 *(2 Punti)*

// -- Codice 3:
class Node<T> { /* ... */ }
Node<Circle> nc = new Node<Shape>(); errore qui
Node<Circle> ns = nc; errore istruzione 2 *(1 Punto)*

// -- Codice 4:
class Node<T> { /* ... */ }
Node<Shape> nc = new Node<Circle>();errore qui
Node<Shape> ns = nc; errore istruzione 2 *(1 Punto)*

// -- Codice 5:
class Node<T> { /* ... */ }
Node<?> nc = new Node<?>(); errore qui
Node<Shape> ns = nc; errore istruzione 2 *(1 Punto)*

// -- Codice 6:
class Node<T> { /* ... */ }
Node<Circle> nc = new Node<>(); corretto *(1 Punto)*

// -- Codice 7:
class Node<PIPPO> { /* ... */ }
new Node<>(); corretto *(1 Punto)*

// -- Codice 8:
Shape s = null;

circle p sottotipo di Shape

Circle c = s; ~~**corretto *(1 Punto)***~~ errore linea2

// -- Codice 9:

class Node<K> { /* … */ }

Node<Circle> nc = new Node<>();

Node<Circle> ns = nc; **corretto *(1 Punto)***

// -- Codice 10:

class MyList extends ArrayList<Rectangle> { /* … */ }

MyList nc = new MyList(); questo non è un generic!

ArrayList<Rectangle> ns = nc; corretto *(1 Punto)*