

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 seguenti 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 seguenti 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)

S
L
S

SSL

Data la seguente classe e gli oggetti definiti come segue

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

PA(S),
T2, T3, T4,
T5, T6, T7,
T8, T9, T10,
T11, T12,
T13, T14,
T15, T16,
T17, T18,
T19, T20,
T21, T22,
T23, T24,
T25, T26,
T27, T28,
T29, T30,
T31, T32,
T33, T34,
T35, T36,
T37, T38,
T39, T40,
T41, T42,
T43, T44,
T45, T46,
T47, T48,
T49, T50,
T51, T52,
T53, T54,
T55, T56,
T57, T58,
T59, T60,
T61, T62,
T63, T64,
T65, T66,
T67, T68,
T69, T70,
T71, T72,
T73, T74,
T75, T76,
T77, T78,
T79, T80,
T81, T82,
T83, T84,
T85, T86,
T87, T88,
T89, T90,
T91, T92,
T93, T94,
T95, T96,
T97, T98,
T99, T100,
T101, T102,
T103, T104,
T105, T106,
T107, T108,
T109, T110,
T111, T112,
T113, T114,
T115, T116,
T117, T118,
T119, T120,
T121, T122,
T123, T124,
T125, T126,
T127, T128,
T129, T130,
T131, T132,
T133, T134,
T135, T136,
T137, T138,
T139, T140,
T141, T142,
T143, T144,
T145, T146,
T147, T148,
T149, T150,
T151, T152,
T153, T154,
T155, T156,
T157, T158,
T159, T160,
T161, T162,
T163, T164,
T165, T166,
T167, T168,
T169, T170,
T171, T172,
T173, T174,
T175, T176,
T177, T178,
T179, T180,
T181, T182,
T183, T184,
T185, T186,
T187, T188,
T189, T190,
T191, T192,
T193, T194,
T195, T196,
T197, T198,
T199, T200,
T201, T202,
T203, T204,
T205, T206,
T207, T208,
T209, T210,
T211, T212,
T213, T214,
T215, T216,
T217, T218,
T219, T220,
T221, T222,
T223, T224,
T225, T226,
T227, T228,
T229, T230,
T231, T232,
T233, T234,
T235, T236,
T237, T238,
T239, T240,
T241, T242,
T243, T244,
T245, T246,
T247, T248,
T249, T250,
T251, T252,
T253, T254,
T255, T256,
T257, T258,
T259, T260,
T261, T262,
T263, T264,
T265, T266,
T267, T268,
T269, T270,
T271, T272,
T273, T274,
T275, T276,
T277, T278,
T279, T280,
T281, T282,
T283, T284,
T285, T286,
T287, T288,
T289, T290,
T291, T292,
T293, T294,
T295, T296,
T297, T298,
T299, T300,
T301, T302,
T303, T304,
T305, T306,
T307, T308,
T309, T310,
T311, T312,
T313, T314,
T315, T316,
T317, T318,
T319, T320,
T321, T322,
T323, T324,
T325, T326,
T327, T328,
T329, T330,
T331, T332,
T333, T334,
T335, T336,
T337, T338,
T339, T340,
T341, T342,
T343, T344,
T345, T346,
T347, T348,
T349, T350,
T351, T352,
T353, T354,
T355, T356,
T357, T358,
T359, T360,
T361, T362,
T363, T364,
T365, T366,
T367, T368,
T369, T370,
T371, T372,
T373, T374,
T375, T376,
T377, T378,
T379, T380,
T381, T382,
T383, T384,
T385, T386,
T387, T388,
T389, T390,
T391, T392,
T393, T394,
T395, T396,
T397, T398,
T399, T400,
T401, T402,
T403, T404,
T405, T406,
T407, T408,
T409, T410,
T411, T412,
T413, T414,
T415, T416,
T417, T418,
T419, T420,
T421, T422,
T423, T424,
T425, T426,
T427, T428,
T429, T430,
T431, T432,
T433, T434,
T435, T436,
T437, T438,
T439, T440,
T441, T442,
T443, T444,
T445, T446,
T447, T448,
T449, T450,
T451, T452,
T453, T454,
T455, T456,
T457, T458,
T459, T460,
T461, T462,
T463, T464,
T465, T466,
T467, T468,
T469, T470,
T471, T472,
T473, T474,
T475, T476,
T477, T478,
T479, T480,
T481, T482,
T483, T484,
T485, T486,
T487, T488,
T489, T490,
T491, T492,
T493, T494,
T495, T496,
T497, T498,
T499, T500,
T501, T502,
T503, T504,
T505, T506,
T507, T508,
T509, T510,
T511, T512,
T513, T514,
T515, T516,
T517, T518,
T519, T520,
T521, T522,
T523, T524,
T525, T526,
T527, T528,
T529, T530,
T531, T532,
T533, T534,
T535, T536,
T537, T538,
T539, T540,
T541, T542,
T543, T544,
T545, T546,
T547, T548,
T549, T550,
T551, T552,
T553, T554,
T555, T556,
T557, T558,
T559, T560,
T561, T562,
T563, T564,
T565, T566,
T567, T568,
T569, T570,
T571, T572,
T573, T574,
T575, T576,
T577, T578,
T579, T580,
T581, T582,
T583, T584,
T585, T586,
T587, T588,
T589, T590,
T591, T592,
T593, T594,
T595, T596,
T597, T598,
T599, T600,
T601, T602,
T603, T604,
T605, T606,
T607, T608,
T609, T610,
T611, T612,
T613, T614,
T615, T616,
T617, T618,
T619, T620,
T621, T622,
T623, T624,
T625, T626,
T627, T628,
T629, T630,
T631, T632,
T633, T634,
T635, T636,
T637, T638,
T639, T640,
T641, T642,
T643, T644,
T645, T646,
T647, T648,
T649, T650,
T651, T652,
T653, T654,
T655, T656,
T657, T658,
T659, T660,
T661, T662,
T663, T664,
T665, T666,
T667, T668,
T669, T670,
T671, T672,
T673, T674,
T675, T676,
T677, T678,
T679, T680,
T681, T682,
T683, T684,
T685, T686,
T687, T688,
T689, T690,
T691, T692,
T693, T694,
T695, T696,
T697, T698,
T699, T700,
T70

PAI SC²

```
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
11 struct C {
12     C() { cout << "C+"; }
13     ~C() { cout << "C-"; }
14 };
15
16 struct X : A, B, protected C {
17     X() { cout << "X+"; }
18     ~X() { cout << "X-"; }
19 };
```

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)

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)

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

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 }
10 class 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'output 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):

- ☒ 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)
- ☒ static void prova5(String s, int ...i){ /**/ } (Selezionato = 1 Punto, Non selezionato = 0 Punti)

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)
```

// -- Codice 2:

```
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>();

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

// -- Codice 4:

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

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

Node<Shape> ns = nc; errore istruzione 2 (1 Punto)

// -- Codice 5:

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

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

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<PIPP0> { /* ... */ }

new Node<>(); corretto (1 Punto)

// -- Codice 8:

Shape s = null;

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();

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