Programmazione 2
Prima Prova Parziale – 21.10.2002

Nome..... Cognome.....

Matricola..... Laurea in.....

Tempo a disposizione: 2H**. Non si possono consultare appunti e libri. Ogni quesito a risposta multipla ha esattamente una risposta corretta: segnare con una croce la risposta scelta. Una risposta corretta vale 1, una risposta errata viene penalizzata con -0.5, nessuna risposta vale 0. Dove previsto scrivere CHIARAMENTE la risposta nell'apposito spazio.**

1. Si consideri il seguente programma.

```
#include<iostream>

class A {
private:
  int x;
public:
  A(int k=0): x(k) {};
  A operator+(A a) {A aux(x + a.x); return aux;};
  operator int() {return x;};
  A F(){++x; return *this;};
};


class C {
public:
  static A a;
  static void G(A& x, A& y){ a = a + x.F() + y.F();}
};
A C::a(1);

main() {
  A p(2), q, r;
  C::G(p,q); cout << C::a << ' ';
  C::G(q,r); cout << C::a << ' ';
  C::G(p,r); cout << C::a;
}
```

   (a) stampa in output `5 7 11`
   (b) stampa in output `4 6 10`
   (c) stampa in output `5 8 14`
   (d) stampa in output `4 7 13`

2. Si considerino le seguenti istruzioni:

```
vector<int> v;
v.push_back(3);
```

   Quale delle seguenti affermazioni è vera?

   (a) è invocato il metodo statico `push_back` sul vector `v`
   (b) la funzione esterna `push_back` è invocata sul vector `v`
   (c) il metodo `push_back` inizializza il vector `v`
   (d) è invocato il metodo `push_back` sul vector `v`

3. Il seguente programma compila.

```
#include<iostream>
#include<vector>

class B {
public:
  vector<int> v;
  B(int x=1, int y=2) {v.push_back(x); v.push_back(y); cout << "B012 ";};
  B(const B& obj): v(obj.v) {cout << "Bc "; }
};

class C {
private:
  B t;
  B* p;
  B s;
public:
  vector<int> w;
  C(int x=0, B y = B(3)): s(y), w(y.v) {
    w.push_back(x);
    cout << *(w.end()-1) << " C012 ";
  };
};

B F(B x, C& y) {
  (x.v).push_back((y.w)[0]);
  return x;
};

main() {
  B b(6); cout << "UNO\n";
  C c1(3,b); cout << "DUE\n";
  C c2; cout << "TRE\n";
  b=F(b,c2); cout <<"QUATTRO\n";
  for(vector<int>::iterator i = (b.v).begin(); i!=(b.v).end(); i++)
    cout << *i << " ";
  cout << "CINQUE";
}
```

Cosa stampa in output?

..... UNO

..... DUE

..... TRE

..... QUATTRO

..... CINQUE

4. Si consideri il seguente programma.

```
#include<iostream>

class C {
public:
  int x;
  C(int k=5): x(k) {};
  C* m(C& c) {
    if((c.x != 5) && (x==5)) return &c;
    return this;
  };
};

main() {
  C a, b(2), c(a);
  cout << (b.m(b))->x << ' ' << (a.m(a))->x << ' ' << (b.m(c))->x
       << ' ' << c.m(a) << ' ' << c.m(c);
}
```

   (a) stampa in output 2 2 5 e di seguito 2 indirizzi diversi

   (b) stampa in output 2 2 5 e di seguito 2 indirizzi uguali

   (c) stampa in output 2 5 2 e di seguito 2 indirizzi uguali

   (d) non compila

5. Si consideri il seguente programma.

```
#include<iostream>
#include<string>

class C {
private:
  int d;
public:
  C(string s=""): d(s.size()) {};
  explicit C(int n): d(n) {};
  operator int() {return d;};
  C operator+(C x) {return C(d+x.d);};
};

main() {
  C a, b("pippo"), c(3);
  cout << a << ' ' << 1+b << ' ' << c+4 << ' ' << c+b;
}
```

   (a) non compila

   (b) stampa in output: .....

6. Si completi la frase: “Il costruttore di default standard ...” in modo da ottenere un’affermazione vera:

   (a) “... è sempre disponibile in ogni classe”

   (b) “... è disponibile quando non è definito esplicitamente il costruttore di default”

   (c) “... è disponibile quando non ha argomenti”

   (d) “... è disponibile se non sono stati definiti esplicitamente altri costruttori”

7. Il seguente programma compila.

```
#include<iostream>

class C {
public:
  C(int x=1): dim(x), a(new char[x]) {for(int i=0; i<dim; i++) a[i]='x';};
  C& operator=(const C& c) {
    if(this != &c) {
      delete[] a; dim = c.dim; a = new char[dim];
      for(int i=0; i<dim; i++) a[i]=(c.a)[i];
    }
    return *this;
  };
  void F(int i, char c) {a[i]=c;};
  void print() {for(int i=0; i<dim; i++) cout << a[i]; cout << ' ';};
private:
  int dim;
  char* a;
};

main(){
  C x(4), y(x), z;
  z=y; z.F(2,'h'); y.F(0,'p'); x.F(3,'w');
  x.print(); y.print(); z.print(); cout << "UNO\n";
  C v=y; v.F(2,'k'); x.F(1,'t');
  x.print(); y.print(); z.print(); v.print(); cout << "DUE\n";
}
```

Cosa stampa in output?

..... UNO

..... DUE

8. Definire, separando interfaccia ed implementazione, una classe `Data` i cui oggetti rappresentano una data con giorno della settimana (lun-mar-...-dom). La classe deve includere:

   (a) opportuni costruttori

   (b) metodi di selezione per ottenere giorno della settimana, giorno, mese, anno di una data

   (c) l'overloading dell'operatore di output `<<` esternamente alla classe

   (d) l'overloading dell'operatore di uguaglianza `==`

   (e) l'overloading dell'operatore relazionale `<` che ignori il giorno della settimana

   (f) un metodo `aggiungi_uno` che avanza di un giorno la data di invocazione. Esempi: lun 21/10/2002 ⇒ mar 22/10/2002; gio 31/1/2002 ⇒ ven 1/2/2002; mar 31/12/2002 ⇒ mer 1/1/2003. Ignorare gli anni bisestili

   Esemplificare l'uso della classe e di tutti i suoi metodi tramite un esempio di `main()`.