NOTE SULLO SVOLGIMENTO DELLA PROVA SCRITTA:

- **SPEGNERE I TELEFONINI;**
- **SCRIVERE IL PROPRIO NOME, COGNOME E NUMERO DI MATRICOLA SU OGNI FOGLIO UTILIZZATO;**
- **NON È POSSIBILE CONSULTARE NESSUN TIPO DI MATERIALE;**
- **NON È POSSIBILE UTILIZZARE CALCOLATRICI;**
- **PRIMA DI SCRIVERE LA SOLUZIONE DELL'ESERCIZIO, INSERIRE IL NUMERO DI ESERCIZIO CHE SI STA RISOLVENDO. PER ESEMPIO, SCRIVERE "ESERCIZIO N. 1" QUANDO SI STA RISOLVENDO L'ESERCIZIO N. 1;**
- NON COPIARE DAL VICINO (NON È DETTO CHE IL VICINO SIA PIÙ BRAVO DI VOI);
- NON PERMETTETE AL VICINO DI COPIARE (È SPIACEVOLE VEDERSI ANNULLARE IL COMPITO SENZA COLPE);
- TUTTI GLI ESERCIZI VALGONO 5 PUNTI
- **ALLA FINE DELLA PROVA, RICONSEGNARE TUTTI I FOGLI UTILIZZATI.**

1) Si mostri l'uscita a video del programma C++ seguente:

```
#include <iostream>
using namespace std;

class A
{
protected:
      int x;
public:
      A() {x = 1;  cout << "A default : " << x <<  endl;}
      A(int k) { x = k; cout << "A :" << k  << endl;}
      virtual void stampa(){cout << "stampa A:" << x << endl;}
      ~A(){cout << "via A" << endl; }
};

class B:  public A
{
public:
      B() : A(20) {x++; cout << "B default"  << endl;}
      B(int n) { x += n ; cout << "B: " << n  << endl;}
      void stampa(){cout << "stampa B: " << x << endl;}
      virtual ~B(){cout << "via B" << endl;}
};

class C : public B
{
      int x;
public:
       C(){ x = 4; cout << "C default" << endl;}
       C(int n) {x=n ; cout << "C : " << n  << endl;}
       void stampa() {cout << "stampa C:" << x << endl;}
      ~C(){cout << "via C" << endl;}
};
```

```
int main()
{     cout << "(1)" << endl;
      A* p1 = new B(3);
      B* p2 = new C(10);
      cout << "(2)" << endl;
      p1->stampa();
      cout << "(3)" << endl;
      p2->stampa();
      cout << "(4)" << endl;
      delete p1;
      cout << "(5)" << endl;
      delete p2;
       return 0;
}
```

2) Si mostri l'uscita a video delle esecuzioni del programma C++ seguente con, rispettivamente, gli ingressi:

```
#include<iostream>
using namespace std;

template<class T, int D = 3>
class A
{
      T vettore[2];
public:
      A (T x1, T x2)
      {     vettore[0] = x1;
            vettore[1] = x2;
      }
      T f () {return vettore[0] + vettore[1] + D; }
};

template<class T1, class T2>
void funzione (T1 y1, T2 y2)
{     A<T1> obj1 (y1, y1);
      A<T2, 1> obj2 (y2, y2);
      cout << obj1.f() <<  endl;
      cout <<  obj2.f() <<  endl;
}
```

```
int main ()
{
    cout << "(1)" << endl;
    funzione (2, 5);
    cout << "(2)" << endl;
    funzione (3.4, 20);
    cout << "(3)" << endl;
    funzione<int>(3.4, 20);
    return 0;
}
```

3) Data la rappresentazione $(1\ 2\ 2\ 1)_3$ in base 3, trasformarla in base 7.
Data la rappresentazione in complemento a due (11010110), esprimere il numero in base 10.

4) Scrivere una funzione ricorsiva che dato un intero N produca la stampa seguente (nell'esempio, N = 5); nella definizione della funzione utilizzare il numero di argomenti ritenuti necessari.

5 4 3 2 1 0
– 5 4 3 2 1
– – 5 4 3 2
– – – 5 4 3
– – – – 5 4
– – – – – 5

5) Sia data la struttura seguente

```
struct elem{ char* nome; elem* pun;};
```

Scrivere una funzione che, data una lista `l` di elementi di tipo `elem`, ordinata per valori crescenti del campo `nome`, apra in lettura un file il cui nome è passato come argomento alla funzione, legga fino alla fine del file stringhe (composte al massimo di 20 caratteri) e le inserisca nella lista in modo da mantenerla ordinata.

6) Data una matrice di interi composta di `r` righe e `c` colonne, con `r` e `c` variabili, scrivere una funzione che stampa su uscita standard la riga la cui somma degli elementi è la maggiore e la riga la cui somma degli elementi è la minore. Nella matrice seguente, la funzione deve stampare le righe [6 7 3 4] e [2 3 1 2].

3 5 6 4
2 3 1 2
6 7 3 4