

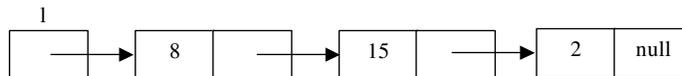
NOTE SULLO SVOLGIMENTO DELLA PROVA SCRITTA:

- **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);**
- **I PRIMI TRE ESERCIZI VALGONO 6 PUNTI. I RIMANENTI TRE VALGONO 4 PUNTI.**
- **ALLA FINE DELLA PROVA, RICONSEGNARE TUTTI I FOGLI UTILIZZATI.**

1) Sia data la struttura elem seguente:

```
struct elem {int info; elem * pun; };
```

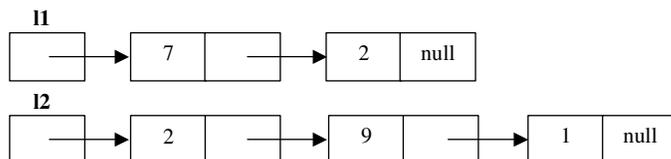
Scrivere una funzione ricorsiva che, data una lista di elementi di tipo elem passata come argomento alla funzione, restituisca la somma dei campi info degli elementi che si trovano in posizione dispari visitando la lista dalla testa alla coda. Supporre che il primo elemento si trovi in posizione 1 della lista. Per esempio, se la funzione viene chiamata con la lista l seguente, la funzione restituirà 10.



2) Sia data la struttura elem seguente:

```
struct elem {int info; elem * pun; };
```

Scrivere una funzione che, date due liste l1 e l2 di elementi di tipo elem, restituisca una lista ordinata per valori crescenti del campo info che contiene tutti gli elementi delle liste l1 e l2. Per esempio, se la funzione viene chiamata con le liste l1 e l2 seguenti,



la funzione restituisce la lista l3 seguente:



3) Scrivere una funzione che, dato un vettore vett di $N > 0$ elementi di tipo intero e un intero $k > 0$, restituisca un vettore che contiene gli elementi del vettore vett ruotati a destra di k posizioni. Per esempio se la funzione viene chiamata con il vettore $\text{vett} = [0\ 5\ 0\ 0\ 9\ 1\ 3\ 4]$ e l'intero $k = 3$, la funzione restituisce il vettore: $[1\ 3\ 4\ 0\ 5\ 0\ 0\ 9]$.

4) Data la rappresentazione $(A75)_{16}$ in base 16, trasformarla in base 2.
Rappresentare il numero negativo -37 in complemento a due su 8 bit.

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

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

class A{
public:
    int k;
    A() { k = 5; cout << "nuovo A default " << "k=" << k << endl;}
    A(int n) { k = n; cout << "nuovo A " << "k=" << k << endl; }
    void f() { cout << "A::f(), k=" << k << endl; }
    ~A() { cout << "via A" << endl; }
};

class B: public A{
public:
    B(): A(4) { cout << "nuovo B default " << "k=" << k << endl; }
    B(int n) { k = n; cout << "nuovo B" << "k=" << k << endl; }
    virtual void f() { cout << "B::f(), k=" << k << endl; }
    ~B() { cout << "via B" << endl; }
};

class C: public B{
public:
    int k;
    C(int n = 2) { k=n; cout << "nuovo C " << "k=" << k << endl; }
    void f() { cout << "C::f(), k=" << k << endl; }
    ~C() { cout << "via C" << endl; }
};
```

```
int main(){
{
    C obj(10);
    A* pa = &obj;
    B* pb = &obj;
    obj.f();
    pa->f();
    pb->f();
}
return 0;
}
```

6) Si mostri l'uscita a video del programma C++ seguente con input -1, 0 e 1:

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

void g (int n){
    cout << n << endl;
    try {
        if (n==0) throw 1;
        if (n==1) throw 'c';
        cout << n << endl;
    }
    catch(int) { cout << "ecc. int da g" << endl;}
    cout << "fine g" << endl;
}

int f(int y) {
    g(y);
    return 9;
}
```

```
int main(){
    int x;
    cin >> x;
    try {
        cout << x << endl;
        if (x==0) throw "cccc";
        cout << f(x) << endl;
    }
    catch(char) {cout << "ecc. char da main" << endl;}
    catch(int) { cout << "int da main" << endl;}
    catch(...) { cout << "default da main" << endl;}
    cout << "fine main" << endl;
    return 0;
}
```