Fondamenti di Programmazione 2

Prova d'esame 14 Febbraio 2023

Esercizio 1

Data la seguente porzione di programma rispondere alle domande corrispondenti:

```
#include<iostream>
using namespace std;
int main() {
    int v[] = {...la tua matricola...};
    int *x = v + 2;
    int *y = v + 4;
    *(x+1) = *x;
    *(y+1) = *y;
    // A: Cosa stampa la seguente istruzione?
    for (int i = 0; i < 6; ++i) cout << v[i];
    cout << endl;

    // B: Cosa stampa la seguente istruzione?
    cout << (y - x) + (*y + *x) << endl;

    int *p, *q = new int[10];
    // C: Qual è il modo corretto di deallocare la memoria dinamica?
    /* 1 */ for (int i = 0; i < 10; ++i) { delete q[i]; delete p[i]; }
    /* 2 */ delete[] q; delete[] p;
    /* 3 */ delete p; delete q;
    /* 4 */ delete[] q;

    int &a = v[1];
    int b = v[0];
    b = *y;
    a = *y;
    // D: Cosa stampa la seguente istruzione?
    for (int i = 0; i < 6; ++i) cout << v[i];
    cout << endl;
}
```

Esercizio 2

Consideriamo la seguente classe `Prodotto`, che si può supporre essere implementata:

```
class Prodotto {
public:
    Prodotto(string, int);
    Prodotto(const Prodotto&);
    string get_nome() const;
    double get_prezzo() const;
    bool operator==(const Prodotto&);
private:
    string nome;
    double prezzo;
};
```

Completare opportunamente l'implementazione della classe `ListaDellaSpesa`, inserendo la parte dati necessaria, e completando i metodi sotto riportati. La classe deve permettere di tenere traccia di quali prodotti, e in quali quantità, vogliamo acquistare quando andiamo a fare la spesa.

```
class ListaDellaSpesa {
public:
    /* Inserisce il Prodotto `p` nella lista della spesa, con quantità `q`.
         Se già presente, incrementa la quantità. */
    void inserisci(const Prodotto& p, int q);
    /* Rimuove il prodotto `p` dalla lista della spesa.
     Restituisce `true` se `p` era presente, `false` altrimenti. */
    bool rimuovi(const Prodotto& p);
    /* Restituisce il costo totale della lista della spesa.
     Il costo di un prodotto nella lista è calcolato
     come il prezzo del prodotto moltiplicato la sua quantità
     nella lista. */
    virtual double totale() const;
};
```

Successivamente, sfruttare l'ereditarietà per implementare una classe `ListaDellaSpesaScontata`, il cui metodo `totale` applicherà uno sconto del 75% ad ogni `Prodotto` che viene acquistato con quantità maggiore di 5.

Esercizio 3

Scrivere una funzione che, preso in input un albero binario interi, restituisca `true` se e solo se esiste almeno un nodo foglia x tale che la somma dei valori informativi dei nodi sul percorso dalla radice di T a x è pari a zero, `false` altrimenti.

Esercizio 4

Scrivere una funzione che preso in input un grafo non orientato G e un intero positivo k restituisca `true` se e solo se è possibile scegliere un insieme di nodi W in modo che siano verificate le seguenti condizioni:

- W contenga esattamente k nodi;
- per ogni arco (u, v) di G, è vero che almeno uno dei nodi u, v è stato incluso in W - ma non entrambi;
- la somma del *grado* dei nodi inclusi in W è minore o uguale a n, dove n è il numero di nodi di G

Per grado di un nodo v si intende il numero di archi incidenti in v.