Fondamenti di Programmazione 2
(Corso di Laurea in Informatica)
Esame del 16/09/2025

UNIVERSITÀ DELLA CALABRIA

Esercizio 1

Data la seguente porzione di programma rispondere, dando motivazione, alle domande corrispondenti:

```
int f (int& k) {return k+2}

int main() {
    int* matricola = new int[6] {//Inserire qui la tua matricola};

    // 1. La seguente istruzione è corretta? Se si, cosa stampa?
    cout << *(matricola[2]) << endl;

    // 2. La seguente istruzione è corretta? Se si, che cosa stampa?
    matricola[3] = f(matricola[2]);
    cout << matricola[3] << " " << matricola[2] << endl;

    // 3. Le seguenti istruzioni sono corrette? Motivare la risposta
    float* m = new float(6);
    for (int i=0; i < 6; i+=2) {
       m+=matricola[i] * matricola[i+1];
    }
   cout << m/=4 << endl;

   // 4. Scrivere sul foglio le istruzioni per deallocare la memoria
   dinamica allocata nel programma.

    return 0;
}
```

Esercizio 2

Si richiede di implementare la classe `VettorePazzo`, ereditando opportunamente da `vector<unsigned>`, che memorizzi numeri interi positivi e un `numero_pazzo` ricevuto a tempo di costruzione. Deve permettere almeno l'inserimento in coda e l'accesso per indice tramite operatore `[]`, allo stesso modo di come avviene per un classico `vector`, rispettando inoltre le seguenti condizioni:

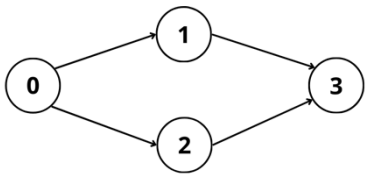
- Se si prova ad inserire il `numero_pazzo` o un suo multiplo, nel vettore non viene inserito nulla;
- Se il vettore contiene un numero di elementi multiplo del `numero_pazzo`, l'operatore `[]` restituirà sempre `0`;
- L'operatore `[]`, inoltre, non deve mai andare in errore. Sia `n` il numero di elementi attualmente presenti nel vettore e `i` l'indice a cui si sta provando ad accedere. L'operatore deve essere implementato secondo il seguente criterio:
  - Se $i < 0$ e $|i| \geq n$, restituisce sempre il primo elemento del vettore;
  - Se $i < 0$ e $|i| < n$, restituisce l'i-esimo elemento da destra;
  - Se $i \geq 0$ e $|i| < n$, restituisce l'i-esimo elemento da sinistra;
  - Se $i \geq 0$ e $|i| \geq n$, restituisce sempre l'ultimo elemento del vettore.
- Due `VettorePazzo`, se confrontati tra di loro, risultano sempre uguali, mentre, per stabilire se un `VettorePazzo` è minore di un altro, si prende in considerazione soltanto il loro `numero_pazzo`.

Fondamenti di Programmazione 2
(Corso di Laurea in Informatica)
Esame del 16/09/2025

UNIVERSITÀ DELLA CALABRIA

Esercizio 3

Sia `g` un grafo orientato e `x` e `y` due suoi nodi. Essi sono sovrapponibili se hanno esattamente gli stessi archi entranti ed esattamente gli stessi archi uscenti (*nella figura in basso, i nodi 1 e 2 sono sovrapponibili perché entrambi hanno un solo arco uscente, verso 3 e un solo arco entrante, che arriva da 0*). Scrivere una funzione che, dato un `Grafo g`, (implementato tramite la classe `Grafo` vista a lezione), restituisca tutte le coppie di nodi tra di loro sovrapponibili.



La classe `Grafo` mette a disposizione la seguente interfaccia pubblica di metodi costanti (con `g` istanza della classe):

- `g.n()` restituisce il numero di nodi del grafo,
- `g.m()` restituisce il numero di archi del grafo,
- `g(i,j)` restituisce `true` se esiste l'arco diretto tra il nodo `i` e il nodo `j`.

I nodi sono etichettati con i valori da `0` a `g.n()-1`.

Esercizio 4

Nel gioco degli scacchi, la regina può muoversi di un numero arbitrario di celle in orizzontale, verticale o diagonale, mentre l'alfiere si muove, sempre di un numero arbitrario di celle, ma solamente in diagonale. Si dice inoltre che un pezzo ne "minaccia" un altro se può raggiungerne la posizione con una sola mossa attraversando solo celle libere.
Si richiede di scrivere un programma che, su una scacchiera quadrata `n x n` (con `n` dato in input) su cui sono già posizionati alcuni alfieri (le cui posizioni sono date in input tramite un `vector<pair<unsigned,unsigned>>`), posizioni `k` regine in modo tale che:

- nessuna regina sia minacciata da un'altra regina;
- nessuna regina sia minacciata da un alfiere;
- nessuna cella sia occupata da più di un pezzo;

Deve anche valere che, dopo aver posizionato tutte le `n` regine, tutti gli alfieri devono essere minacciati. Si noti che è possibile che qualche regina non minacci alcun alfiere, come è possibile che una stessa regina ne minacci più di uno. Se per l'istanza del problema in input esiste una soluzione, il programma dovrà stamparla, stampando "IMPOSSIBILE" altrimenti.