Programmazione ad Oggetti
Traccia Laboratorio 22 novembre 2019

Esercizio 1.
Realizzare la classe Prenotazione che contiene almeno i seguenti campi:

- int numero;
- int posti;
- string codice;

dove è possibile avere 10 codici di prenotazioni: “a0001”, “b0002”, “c0003”, “d0004”, “e0005”, “f0006”, “g0007”, “h0008”, “i0009”, “z0010”. La scelta dei metodi da inserire nella classe Prenotazione è libera.

Realizzare la classe GestorePrenotazioni che permetta di gestire le prenotazioni. La classe deve contenere al suo interno un campo di tipo list<Prenotazione*>.

a) bool aggiungiPrenotazione(int posti, string codice); aggiunge una prenotazione alla lista di prenotazioni.
   Una prenotazione può essere aggiunta solo se c’è abbastanza posto per quella determinata prenotazione, dove il numero di posti disponibili è stabilito dagli ultimi due numeri del codice della prenotazione. Ad esempio, c’è un solo posto disponibile per il codice “a0001” e 4 posti disponibili per il codice “d0004”.

   Se la prenotazione può essere aggiunta, il metodo crea una nuova prenotazione e l’aggiunge all’interno della lista. Al momento dell’aggiunta deve essere generato un numero casuale X non assegnato a nessuna altra prenotazione nella lista.

   Se la prenotazione non può essere aggiunta il metodo restituisce false, altrimenti true.

b) bool rimuoviPrenotazione(int numero); rimuove una prenotazione dalla lista. Restituisce true se la prenotazione è presente, false altrimenti.
c) void stampaPrenotazioni(string codice) const; stampa tutte le prenotazioni con quel codice.
d) void ordinaPrenotazioni(); ordina le prenotazioni come segue:
   - Prima tutte le prenotazioni il cui codice inizia con la lettera a (non importa l’ordine).
   - Poi tutte le prenotazioni il cui codice inizia con la lettera b (non importa l’ordine).
   - Poi tutte le prenotazioni il cui codice inizia con la lettera c (non importa l’ordine).
   - Infine, tutte le altre prenotazioni in ordine di numero.

Infine, realizzare un main di prova che permetta l’interazione con la classe GestorePrenotazioni. La scelta del menu è libera, ma deve essere possibile usare tutti i metodi della classe GestorePrenotazioni.

CONTINUA A PAGINA 2.

Esercizio 2.
Realizzare una classe CodaPrioritaria che gestisca la coda di persone in fila. La classe deve contenere un campo privato list<int> e i seguenti metodi:

a) void aggiungi(int n); se n è 0 il numero non deve essere aggiunto.
b) int prossimoNumero() const;
c) void rimuoviProssimoNumero();
d) int size() const;

La coda ha il seguente ordine di priorità:

- i numeri positivi precedono i negativi
- tra i numeri positivi, i pari precedono i dispari
- tra i numeri negativi, i dispari precedono i pari
- a parità delle altre condizioni, il numero più piccolo (in valore assoluto) precede il più grande.

Ad esempio, un possibile ordine è:
2 4 1 3 5 -1 -3 -6 -8

Esercizio 3.
Realizzare la classe Partita che contenga i seguenti campi privati: string squadra1; string squadra2, int golSquadra1; int golSquadra2; string arbitro.

Realizzare la classe Mondiale contenente un campo privato list<Partita> e i seguenti metodi:

- void aggiungi(Partita); aggiunge un elemento all'interno della lista.
- int metodo1(); si consideri come M la media dei gol segnati in tutte le partite (utilizzare la divisione intera). Si consideri come P la partita con meno gol segnati tra quelle con un numero di gol segnati >= M (nel caso in cui più partite soddisfino la condizione prendere la prima in ordine di apparizione nella lista). Il metodo restituisce i gol della partita P. Se non sono presenti partite, restituire -1.
- int metodo2(); restituisce il numero di arbitri che non hanno mai arbitrato più di una partita della stessa squadra. Se non sono presenti partite, restituire -1.
- int metodo3(); restituisce il numero di arbitri che hanno arbitrato solo partite finite con più di 2 gol di differenza. Se non sono presenti partite, restituire -1.
- int metodo4(); Data la seguente definizione:
  Una squadra S1 domina una squadra S2 se:
  - S1 vince una partita in trasferta contro S2 con almeno due gol di scarto, oppure
  - S1 domina S3 e S3 domina S2.

  Il metodo restituisce il numero di squadre dominate dalla squadra1 della prima partita della lista.
  Note: Una squadra S domina sempre se stessa.

  Nel caso in cui non ci siano partite restituire -1.

Dipartimento di Matematica
e Informatica
Università della Calabria
Via P. Bucci, Cubo 30B
87036 Rende (Cs)

Tel. (+39) 0984.49.6402 - 6400
Fax (+39) 0984.49.6410
direttore.demacs@unical.it
dipartimento.demacs@pec.unical.it

www.unical.it