Analisi e progettazione del software — Prova parziale del 25 novembre 2020

Esercizio 1 (punti 8) Si consideri la classe `Data` come definita a lezione (con i campi `giorno`, `mese` e `anno`, i metodi, gli operatori e i costruttori spiegati in classe). Si assuma inoltre disponibile la funzione `int Random(int a, int b)` che restituisce un numero casuale tra `a` e `b` (estremi inclusi), con probabilità uniforme.

Si scriva un costruttore della classe `Data` che riceva come parametro un intero che rappresenta un anno e costruisca un oggetto che abbia una data casuale all'interno dell'anno stesso. *Tutte le date dell'anno devono essere scelte con la stessa identica probabilità.*

Esercizio 2 (punti 10) Un file contiene la sequenze delle temperature registrate in un insieme di date. Ciascuna riga del file è composta dalla data, dal numero di misurazioni effettuate in quel giorno e dalla sequenza delle misurazioni (separate da virgole). Come esempio si consideri il seguente file.

```
7/3/2020 7 4.5, 9.8, 9.2, 7.2, 4.4, 3.2, 4.3
15/5/2020 6 20.1, 22.7, 24.1, 25.2, 26.4, 26.2
23/8/2020 5 28.1, 29.8, 29.2, 32.2, 30.4
```

Si scriva una funzione che prenda come parametro il nome di un file siffatto e restituisca la data con la massima escursione termica. La funzione deve restituire una coppia formata dalla data e dal vettore di tutte le temperature registrate in quella data. Nell'esempio, la funzione deve restituire la coppia $\langle \texttt{7/3/2020}, \{\texttt{4.5}, \texttt{9.8}, \texttt{9.2}, \texttt{7.2}, \texttt{4.4}, \texttt{3.2}, \texttt{4.3}\}\rangle$, essendo l'escursione del giorno `7/3/2020`, pari a 6.6 $(9.8 - 3.2)$, la massima del file.

Si consideri disponibile la funzione `Estremo` che prende come parametri un vettore di reali e un booleano e restituisce il massimo o il minimo valore del vettore, a seconda se il booleano passato come secondo parametro sia `true` o `false`, rispettivamente.

Esercizio 3 (punti 14) Si considerino le classi `A` e `B` e la funzione `main` definite di seguito.

```
class B
{public:
  B(int n) { num = n; }
  void Set(int n) { num = n; }
  char Get() const { return num; }
 private:
  int num;
};
```

```
class A
{public:
  A(int v = 0) { p1 = new B(v); p2 = p1; }
  void Set1(int v) {  p1->Set(v); }
  void Set3(int v) {  p2->Set(v); }
  int Get1() const { return p1->Get(); }
  int Get2() const { return p2->Get(); }
  void Flip() { if (p1 == p2) p2 = new B(p1->Get());
                else { delete p2; p2 = p1; } }
 private:
  B* p1;
  B* p2;
};
```

```
int main()
{
  A a1(5);
  a1.Flip();
  a1.Set1(6);
  A a2 = a1;
  a2.Set1(4);
  cout << a1.Get1() << '/'
       << a1.Get2() << endl;
  return 0;
}
```

- Riportare cosa stampa il programma e cosa invece stamperebbe in assenza di interferenza (mostrando anche il procedimento).
- Scrivere il costruttore di copia, l'operatore di assegnazione e il distruttore della classe `A` in modo che evitino la condivisione di memoria e rilascino la memoria dinamica non più utilizzata.