Prova del 2021-febbraio-15

Soluzioni

Prova 1 – esercizio 1

Data la definizione seguente per realizzare liste di elementi interi:

```
typedef struct nodo
{int value ;
struct nodo * next ;} NODO ;
typedef NODO * list ;
```

si scriva una funzione iterativa con prototipo:

```
int sumEq(list L, int x)
```

che restituisce la somma dei valori memorizzati in una lista `list L` uguali a un valore intero `int x` passato alla funzione.

Ispiriamoci al programma che calcola la lunghezza di una lista (iterativa)

```
int length(list L)
{ int lung=0 ;
  while(L!=NULL) {
        lung=lung+1;
        L=L->next; }
  return lung;
}
```

Prova 1 – esercizio 1 – soluzione iterativa

Non sommo 1 ma x ogni volta che il nodo corrente ha valore uguale ad x …

```
int sumEq(list L, int x)
{ int sum=0 ;
  while(L!=NULL) {
        if (L->value==x) sum=sum+x;
        L=L->next; }
  return sum;
}
```

Facciamo una variante al testo dell'esercizio

- Oppure, se si fosse chiesto di scrivere una funzione ricorsiva

ispiriamoci al programma che calcola la lunghezza di una lista (ricorsiva):

```
int length(list L)
{ if(L==NULL) return 0;
  else return 1+length(L->next);
}
```

Prova 1 – esercizio 1 – soluzione ricorsiva

```
int sumEq(list L, int x)
{ if(L==NULL) return 0;
  else
     if (L->value==x) return x+sumEq(L->next,x)
     else return sumEq(L->next,x);
}
```

Facciamo un'altra variante al testo dell'esercizio

- Oppure, se si fosse chiesto di farlo su un albero (sempre ricorsiva)

Prova 1 – esercizio 1 – se albero anziché lista?

Data la definizione seguente per realizzare alberi binari di elementi interi:

```
typedef struct nodo
{int value ;
struct nodo *left, *right;} NODO ;
typedef NODO *tree ;
```

si scriva una funzione con prototipo:

```
int sumEq(tree T, int x)
```

che restituisce la somma dei valori memorizzati in un albero binario `tree T` uguali a un valore intero `int x` passato alla funzione.

Prova 1 – esercizio 1 – soluzione ricorsiva

```
int sumEq(tree T, int x)
{ if(T==NULL) return 0;
  else
     if (T->value==x)
           return x+sumEq(T->left,x)+sumEq(T->right,x);
     else return sumEq(T->left,x)+sumEq(T->right,x);
}
```

Prova 1 – esercizio 2

Sia data la seguente classe astratta `Figura` :

```
abstract class Figura {
     protected int lati ;
     protected int area ;
     public Figura ( int l, int a) {
             this.lati = l;
             this.area = a; }
     abstract public void print();
}
```

Si scriva una classe derivata concreta `Poligono` , che deriva da `Figura` e che implementa il metodo `print()` stampando a video la concatenazione degli attributi.

Prova 1 – esercizio 2 - soluzione

Sia data la seguente classe astratta `Figura` :

```
class Poligono extends Figura {
      public Poligono ( int l, int a) { super(l,a); }
      public void print() {System.out.println(l+a} ; }
}
```