Introduzione alla Programmazione

claudio.lucchese@unive.it

Esercizi

1. Scrivere una funzione che riceve in input 3 parametri “int A, int B, int* max” e che memorizzi nella locazione riferita da max, il valore del massimo tra A e B
2. Scrivere una funzione che riceve in input 2 parametri “int *A, int *B” e che scambi i valori nelle locazioni riferite dai due puntatori
3. Scrivere una funzione che legge da input un intero N, e dopo aver letto altri N numeri interi, restituisca la somma degli N numeri interi
4. Come sopra, scrivere una funzione che dato un intero N e tre puntatori a intero, legga N numeri interi da input e usi i tre puntatori per memorizzare min, max e media

Esercizi

1. Scrivere una funzione che legge N numeri interi positivi da input e restituisce 1 se sono tutti pari o 0 altrimenti
2. Scrivere una funzione che legge N numeri interi da input e restituisce 1 se la sequenza è crescente e 0 altrimenti
3. Scrivere una funzione che legge N numeri interi da input e restituisce 1 se la sequenza è bitonica e 0 altrimenti (una sequenza si dice bitonica se è prima crescente e poi decrescente o viceversa)

Esercizi

1. Scrivere una funzione che legge *al massimo* N numeri interi da input e restituisce 1 se sono tutti pari o 0 altrimenti, ma smette di chiedere ulteriore input non appena legge un numero negativo
2. Scrivere una funzione che legga da input numeri interi finché l'utente non specifica un numero negativo. Letto il numero negativo, la funzione smette di chiedere ulteriore input e restituisce la media dei valori positivi letti
3. Scrivere una funzione con un parametro T che legge da input almeno 3 numeri interi e smette l'esecuzione con un messaggio a schermo non appena la media degli ultimi 3 numeri letti supera una soglia T
4. Scrivere una funzione con un parametri T e N che legge da input N numeri interi e restituisce il numero di volte per cui la somma di 3 numeri letti consecutivamente è superiore a T

Esercizi

1. Run-Length-Encoding è una codifica di compressione si sequenze di numeri interi molto comune che permette di risparmiare spazio nel caso di ripetizioni (ad es. è usata nella fase finale della compressione JPEG)
   Funziona così: data una sequenza di interi in input, produce una sequenza in output di coppie <intero, num_ripetizioni>; es:
   a. input **1,1,2,2,2,3,3,3,3,1,0,0,0,0,0**
   b. output **1**,2,**2**,3,**3**,4,**1**,1,**0**,5

   Scrivere una funzione che legga input una sequenza di interi e scriva in output (durante l'esecuzione e non tutto alla fine) gli elementi della sequenza di output corretta.
   La lettura da input viene interrotta al primo intero negativo che non farà parte della sequenza.

Esercizi

1. Come il precedente, con la differenza che in input viene letta la sequenza compressa, e viene prodotta output la sequenze originale non compressa.

Altri esercizi

- https://www.codewars.com
- https://codefights.com
- https://www.topcoder.com
- https://www.geeksforgeeks.org/

Esercizio

Q2 (1.5 pt.)

Dato il seguente codice:

```
int* f2 (int a, int b, int c) {
  int *x = &a;
  int *y = &b;
  int *z = &c;
  *x = *y + *z;
  return x;
}
```

Qual è l'output del codice seguente e quali affermazioni sono corrette?

```
printf( "%d \n", f2(1,2,3) );
```

- ☐ L'output è 5 pari a `2 + 3`
- ☐ Non si possono sommare i due puntatori nell'istruzione `*y + *z`
- ☐ Non si può scrivere `&a`, `&b`, `&c`, perchè gli argomenti `1`, `2`, `3` sono costanti e non hanno un indirizzo come le variabili
- ☐ Non è corretto restituire l'indirizzo della variabile `a`

Esercizio

Q2 (1.5 pt.)

Dato il seguente codice:

```
int* f2 (int a, int b, int c) {
  int *x = &a;
  int *y = &b;
  int *z = &c;
  *x = *y + *z;
  return x;
}
```

Qual è l'output del codice seguente e quali affermazioni sono corrette?

```
printf( "%d \n", f2(1,2,3) );
```

- ☐ L'output è 5 pari a `2 + 3`
- ☐ Non si possono sommare i due puntatori nell'istruzione `*y + *z`
- ☐ Non si può scrivere `&a`, `&b`, `&c`, perchè gli argomenti `1`, `2`, `3` sono costanti e non hanno un indirizzo come le variabili
- ☒ Non è corretto restituire l'indirizzo della variabile `a`

Esercizio

Q2 (1.5 punti)

Dato il seguente codice:

```
1 void f2 (int *a, int *b) {
2   int *aux = a;
3   a = b;
4   b = aux;
5 }
```

Qual è l'output del codice seguente e quali affermazioni sono corrette?

```
1 int a = 1, b = 2;
2 f2( &a, &b);
3 printf( "%d %d \n", a, b);
```

- ☐ 1 2
- ☐ 2 1 perchè le variabili vengono scambiate
- ☐ l'istruzione `a = b` non compila perché `a` e `b` sono puntatori
- ☐ è più corretto scrivere `int *aux = &a;`

Esercizio

Q2 (1.5 punti)

Dato il seguente codice:

```c
void f2 (int *a, int *b) {
  int *aux = a;
  a = b;
  b = aux;
}
```

Qual è l'output del codice seguente e quali affermazioni sono corrette?

```c
int a = 1, b = 2;
f2( &a, &b);
printf( "%d %d \n", a, b);
```

- ☒ 1 2
- ☐ 2 1 perchè le variabili vengono scambiate
- ☐ l'istruzione `a = b` non compila perché `a` e `b` sono puntatori
- ☐ è più corretto scrivere `int *aux = &a;`