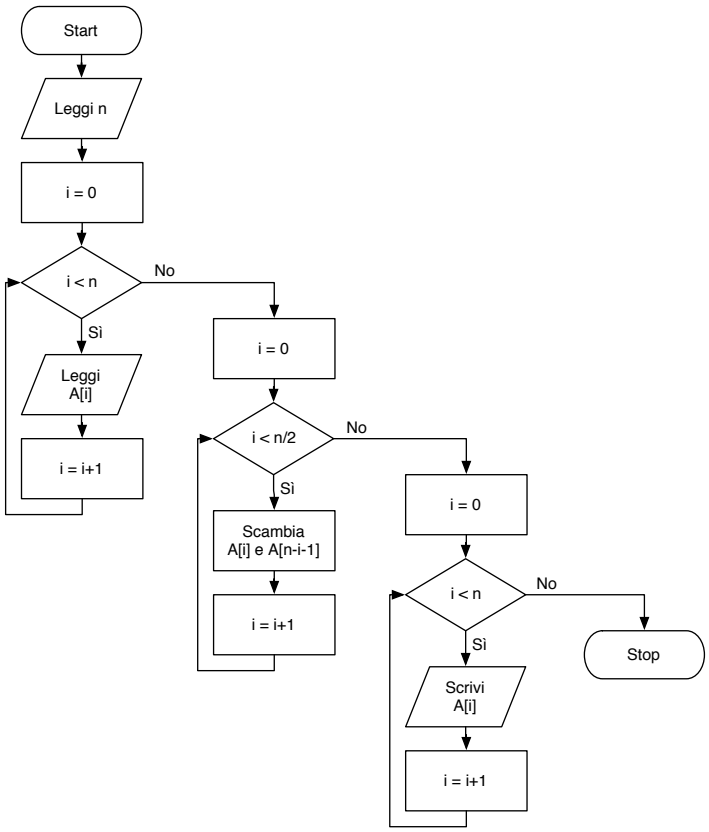
Algoritmi e Strutture Dati (IN110)
Esercitazione n. 2

Marco Liverani *

Esercizio n. 1

Letto in input un array A di n elementi, usando solo questo array, inverte l'ordine degli elementi; ossia se $A = (10, 27, 13, 4)$, allora devo ottenere $A = (4, 13, 27, 10)$; al termine dell'elaborazione stampa l'array A con gli elementi nel nuovo ordine.

Diagramma di flusso



*Università degli Studi Roma Tre, Corso di Laurea in Matematica, Corso di Algoritmi e Strutture Dati (IN110) – sito web del corso http://www.mat.uniroma3.it/users/liverani/IN110/

Pseudo-codifica dell'algoritmo

1: leggi n
2: **per** $i = 0, 1, 2, \ldots, n-1$ **ripeti**
3: leggi A_i
4: **fine-ciclo**
5: $i = 0$
6: **fintanto che** $i < \frac{n}{2}$ **ripeti**
7: scambia i valori delle variabili A_i e A_{n-i-1}
8: $i = i + 1$
9: **fine-ciclo**
10: **per** $i = 0, 1, 2, \ldots, n-1$ **ripeti**
11: scrivi A_i
12: **fine-ciclo**
13: stop

Codifica in linguaggio C

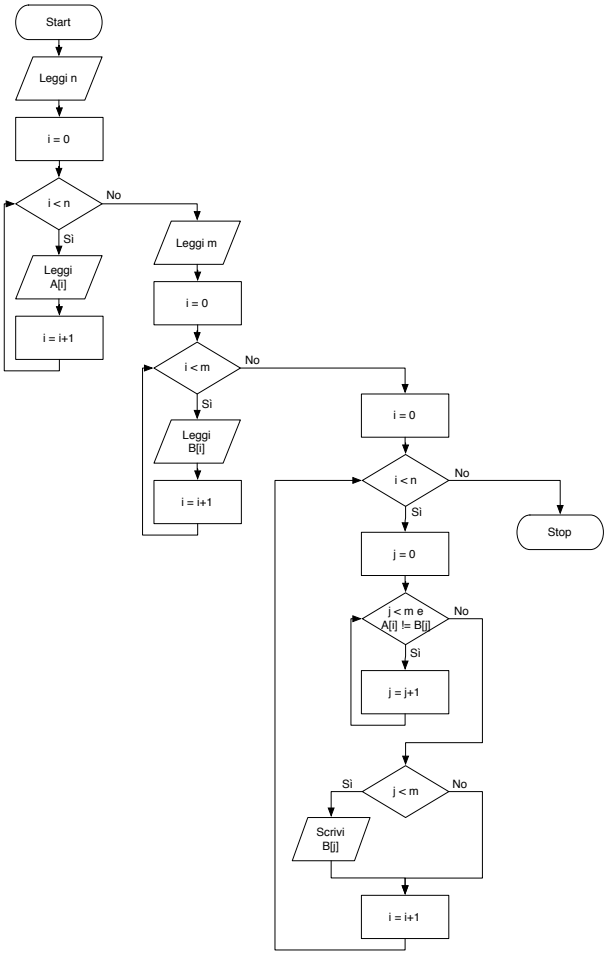
```c
#include <stdlib.h>
#include <stdio.h>
#define MAX 50

int main(void) {
  int i, j, n, x, A[MAX];
  printf("Numero di elementi di A: ");
  scanf("%d", &n);
  printf("Inserisci gli elementi di A: ");
  for (i=0; i<n; i++) {
    scanf("%d", &A[i]);
  }
  for (i=0; i<n/2; i++) {
    x = A[i];
    A[i] = A[n-i-1];
    A[n-i-1] = x;
  }
  for (i=0; i<n; i++) {
    printf("%d ", A[i]);
  }
  printf("\n");
  return(0);
}
```

Esercizio n. 2

Letti in input due array A e B, rispettivamente di n ed m elementi, stampa tutti gli elementi in comune.

Diagramma di flusso



Pseudo-codifica dell'algoritmo

L'algoritmo può essere schematizzato nei seguenti passi di "alto livello":

1: leggi in input l'array A di n elementi
2: leggi in input l'array B di m elementi
3: **per ogni** elemento di A (per ogni A_i, $i = 0, 1, 2, \ldots, n-1$) **ripeti**
4: scorri l'array B fino alla fine oppure fino a quando non avrai trovato un elemento $B_j = A_i$
5: se è stato trovato un elemento di B uguale all'elemento A_i considerato, allora stampa l'elemento B_j
6: **fine-ciclo**

Questi passi possono essere riscritti con maggiore dettaglio ottenendo la seguente pseudo-codifica:

1: leggi n
2: $i = 0$
3: **fintanto che** $i < n$ **ripeti**
4: leggi A_i
5: $i = i + 1$
6: **fine-ciclo**
7: leggi m
8: $i = 0$
9: **fintanto che** $i < m$ **ripeti**
10: leggi B_i
11: $i = i + 1$
12: **fine-ciclo**
13: $i = 0$
14: **fintanto che** $i < n$ **ripeti**
15: $j = 0$
16: **fintanto che** $j < m$ e $A_i \neq B_j$ **ripeti**
17: $j = j + 1$
18: **fine-ciclo**
19: **se** $j < m$ **allora**
20: scrivi B_j
21: **fine-condizione**
22: $i = i + 1$
23: **fine-ciclo**
24: stop

Codifica in linguaggio C

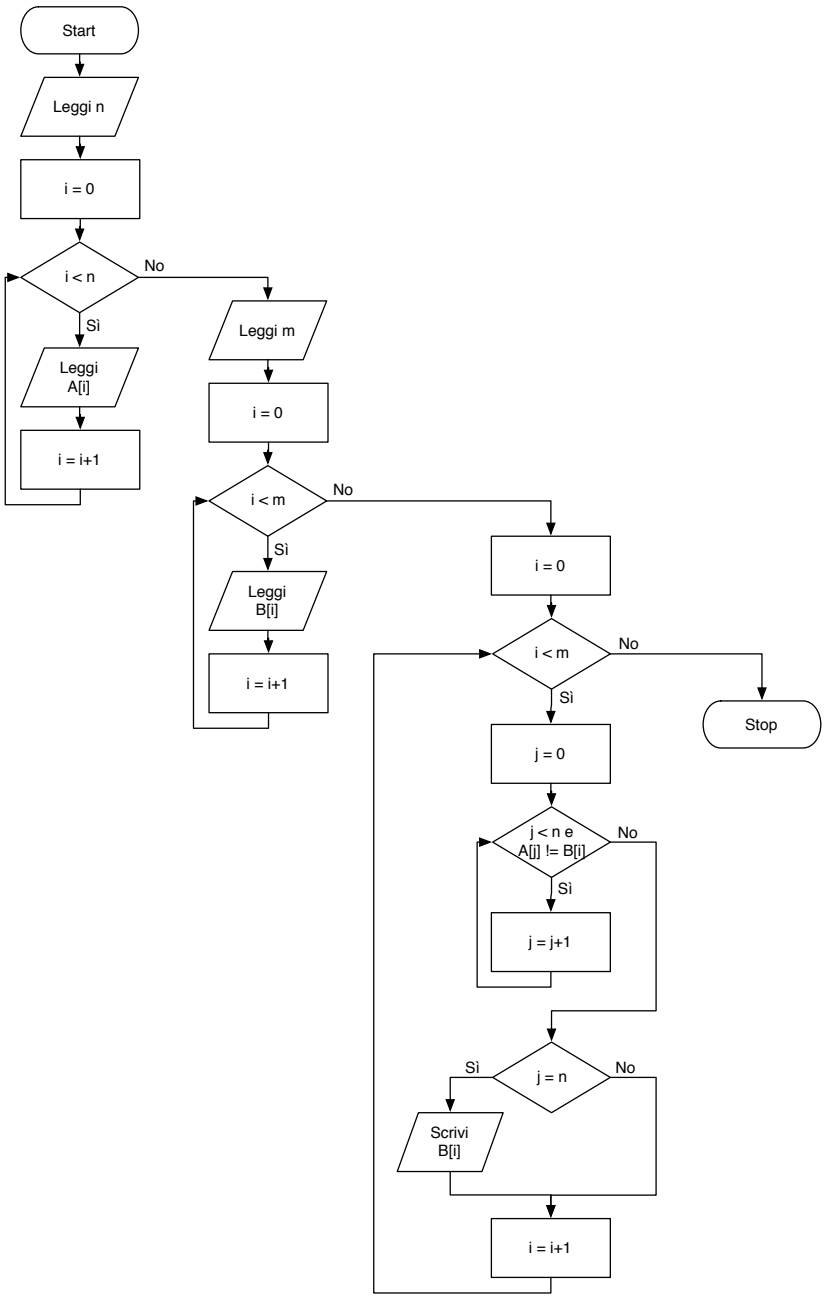
```c
#include <stdlib.h>
#include <stdio.h>
#define MAX 50

int main(void) {
  int i, j, n, m, A[MAX], B[MAX];
  printf("Numero di elementi di A: ");
  scanf("%d", &n);
  printf("Inserisci gli elementi di A: ");
  for (i=0; i<n; i++) {
    scanf("%d", &A[i]);
  }
  printf("Numero di elementi di B: ");
  scanf("%d", &m);
  printf("Inserisci gli elementi di B: ");
  for (i=0; i<m; i++) {
    scanf("%d", &B[i]);
  }
  printf("Elementi comuni ad A e B: ");
  for (i=0; i<n; i++) {
    j = 0;
    while (j < m && A[i] != B[j]) {
      j++;
    }
    if (j < m) {
      printf("%d ", B[j]);
    }
  }
  printf("\n");
  return(0);
}
```

Esercizio n. 3

Letti in input due array A e B, rispettivamente di n ed m elementi, stampa tutti gli elementi presenti in B, ma non in A.

Diagramma di flusso



Pseudo-codifica dell'algoritmo

L'algoritmo risolutivo può essere schematizzato nei seguenti passi elementari:

1: leggi l'array A di n elementi
2: leggi l'array B di m elementi
3: per ogni elemento B_i di B ($i = 0, 1, 2, \ldots, n-1$) ripeti i passi 4-5:
4: scorri l'array A dal primo elemento fino all'ultimo oppure fino a quando non hai trovato un elemento $A_j = B_i$
5: se scorrendo tutto l'array A non hai trovato nessun elemento uguale a B_i allora scrivi B_i

L'algoritmo può essere descritto in modo più puntuale attraverso la seguente pseudo-codifica:

1: leggi n
2: **per** $i = 0, 1, 2 \ldots, n-1$ **ripeti**
3: leggi A_i
4: **fine-ciclo**
5: leggi m
6: **per** $i = 0, 1, 2, \ldots, m-1$ **ripeti**
7: leggi B_i
8: **fine-ciclo**
9: **per** $i = 0, 1, 2, \ldots, m-1$ **ripeti**
10: $j = 0$
11: **fintanto che** $j < n$ e $A_j \neq B_i$ **ripeti**
12: $j = j + 1$
13: **fine-ciclo**
14: **se** $j = n$ **allora**
15: scrivi B_i
16: **fine-condizione**
17: **fine-ciclo**
18: stop

Codifica in linguaggio C

```c
#include <stdlib.h>
#include <stdio.h>
#define MAX 50

int main(void) {
  int i, j, n, m, A[MAX], B[MAX];
  printf("Numero di elementi di A: ");
  scanf("%d", &n);
  printf("Inserisci gli elementi di A: ");
  for (i=0; i<n; i++)
    scanf("%d", &A[i]);
  printf("Numero di elementi di B: ");
  scanf("%d", &m);
  printf("Inserisci gli elementi di B: ");
  for (i=0; i<m; i++)
    scanf("%d", &B[i]);
  printf("Elementi presenti in B ma non in A: ");
  for (i=0; i<m; i++) {
    j = 0;
    while (j<n && A[j] != B[i]) {
      j++;
    }
    if (j==n) {
      printf("%d ", B[i]);
    }
  }
  printf("\n");
  return(0);
}
```