Corso di Algoritmi e Strutture Dati (IN110) – Prof. Marco Liverani – a.a. 2022/2023

Esame scritto del 27 Gennaio 2023 (Appello A)

Si richiede di risolvere entrambi gli esercizi riportando una codifica in linguaggio C completa dei due programmi. Nel caso in cui non si riesca a completare entrambi gli esercizi si suggerisce di riportare almeno la codifica in C delle funzioni principali o una loro pseudo-codifica. È possibile consultare libri e appunti personali, ma non scambiare libri o appunti con altri studenti. I compiti che presenteranno evidenti ed anomale "similitudini" saranno annullati. La prova scritta ha una durata di tre ore, durante le quali non è consentito allontanarsi dall'aula, se non dopo aver consegnato il compito.
Deve essere consegnata solo la "bella copia" del compito scritto; su ciascun foglio deve essere riportato il ***nome****, il* ***cognome*** *e il* ***numero di matricola*** *(o un altro codice identificativo di fantasia) dello studente.*

Esercizio n. 1

Letti in input due numeri interi positivi n e k, generare un vettore A di n numeri interi casuali compresi tra $-k$ e $2k$ (estremi inclusi). Costruire un vettore B di n elementi, tale che B_i sia dato dalla media aritmetica fra il minimo e il massimo dei primi i elementi di A. Stampare in output A e B.

Esempio Siano $n = 10$ e $k = 5$ e sia A il seguente array di numeri casuali costruito come richiesto dall'esercizio: $A = (3, 2, 4, -4, 10, -5, -4, -5, 9, 10)$. Allora il vettore B è il seguente:

$$B = (3.0, 2.5, 3.0, 0.0, 3.0, 2.5, 2.5, 2.5, 2.5, 2.5)$$

Soluzione

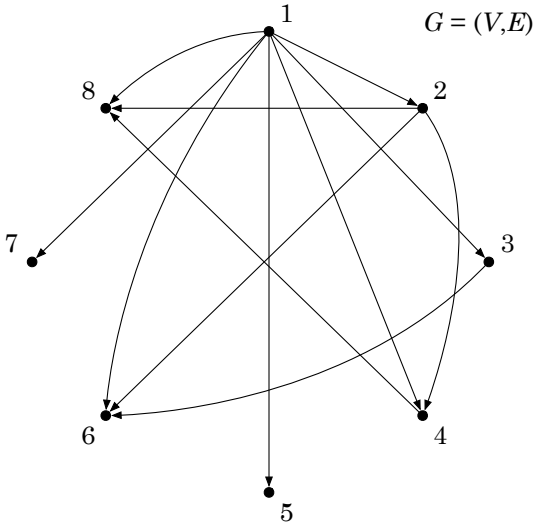
```c
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#define MAX 50

int generaArray(int A[]) {
  int i, k, n;
  srand((unsigned)time(NULL));
  printf("Insersci n e k: ");
  scanf("%d %d", &n, &k);
  for (i=0; i<n; i++)
     A[i] = rand() % (3*k+1) - k;
  return(n);
}

void stampaArrayInteri(int X[], int n) {
  int i;
  for (i=0; i<n; i++)
    printf("%d ", X[i]);
  printf("\n");
  return;
}

void stampaArrayFloat(float X[], int n) {
  int i;
  for (i=0; i<n; i++)
    printf("%f ", X[i]);
```

```
28    printf("\n");
29    return;
30  }
31
32  void costruisciArray(int A[], float B[], int n) {
33    int i, min, max;
34    min = A[0];
35    max = A[0];
36    for (i=0; i<n; i++) {
37      if (A[i] < min)
38        min = A[i];
39      if (A[i] > max)
40        max = A[i];
41      B[i] = (min + max)/2.0;
42    }
43    return;
44  }
45
46  int main(void) {
47    int A[MAX], n;
48    float B[MAX];
49    n = generaArray(A);
50    costruisciArray(A, B, n);
51    stampaArrayInteri(A, n);
52    stampaArrayFloat(B, n);
53    return(0);
54  }
```

Esercizio n. 2

Letto in input un intero n, costruire le liste di adiacenza di un grafo orientato $G = (V, E)$, tale che $V = \{1, 2, \ldots, n\}$ (la numerazione dei vertici inizia da 1) e $(v_i, v_j) \in E(G)$ se e solo se v_j è un multiplo di v_i. Stampare in output le liste di adiacenza di G.

Esempio Sia $n = 8$. Allora il grafo orientato richiesto dall'esercizio è il grafo grafo $G = (V, E)$ rappresentato in figura.



Soluzione

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

struct nodo {
  int info;
  struct nodo *next;
};

void stampaLista(struct nodo *p) {
  while (p != NULL) {
    printf("%d --> ", p->info);
    p = p->next;
  }
  printf("null\n");
  return;
}

void stampaGrafo(struct nodo *V[], int n) {
  for (int i=1; i<=n; i++) {
    printf("%2d: ", i);
    stampaLista(V[i]);
  }
  return;
}
```

```
26
27 int costruisciGrafo(struct nodo *V[]) {
28   int i, j, n;
29   struct nodo *p;
30   printf("Numero di vertici: ");
31   scanf("%d", &n);
32   for (i=1; i<=n; i++) {
33     V[i] = NULL;
34     j = 2*i;
35     while (j <= n) {
36       p = malloc(sizeof(struct nodo));
37       p->info = j;
38       p->next = V[i];
39       V[i] = p;
40       j = j+i;
41     }
42   }
43   return(n);
44 }
45
46 int main(void) {
47   struct nodo *G[MAX];
48   int n;
49   n = costruisciGrafo(G);
50   stampaGrafo(G, n);
51   return(0);
52 }
```