5 5 5 1 6 ...

Numeri casuali

Marco Alberti





Programmazione e Laboratorio, A.A. 2021-2022

Ultima modifica: 17 novembre 2021

Attenzione! Questo materiale didattico è per uso personale dello studente ed è coperto da copyright. Ne sono vietati la riproduzione e il riutilizzo anche parziale, ai sensi e per gli effetti della legge sul diritto d'autore.

Generazione di numeri casuali



La fuzione int rand(void), dichiarata in <stdlib.h>, genera un intero casuale compreso fra 0 e RAND MAX (una costante intera, almeno 32767).

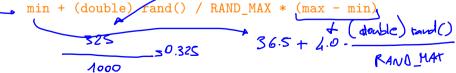
Generazione di int casuali:

(1+ tand() % 6, - compreso for 1 e 6

- L'espressione rand() % 10 vale un numero casuale compreso fra 0 e 9.
- Più in generale, un numero intero casuale compreso fra min e max si ottiene da min + rand() % (max min + 1).

Generazione di double casuali:

- L'espressione (double) rand(), / RAND_MAX vale un double casuale compreso fra 0.0 e 1.0.
- In generale, un double casuale compreso fra min e max si ottiene come



Esercizio



Scrivere un programma che simuli 10 lanci di un dado a 6 facce (cioè stampi 10 numeri casuali compresi fra 1 e 6).

```
097 numeri casuali/dado.c
                                 min=1
  #include <stdio.h>
  #include <stdlib.h>
3
  int main(void) {
    int i;
    for (i = 0; i < 10; i++)
      printf("%d\n", 1 + rand() % 6);
8
    return 0:
                    min
```

Esercizio



Scrivere un programma che simuli 10 rilevazioni di temperature comprese fra 15.0°C e 25.0°C.

```
097 numeri casuali/temperature.c
                              min = 15.0
  #include <stdio.h>
                             Mex = 25.0
  #include <stdlib.h>
3
  int main(void) {
    int i;
    for (i = 0; i < 10; i++) (double)
      printf("%f\n", 15.0 + 10.0 * rand() / RAND_MAX);
sturn 0;
8
    return 0;
                         (max-min)
```

Funzioni per generazione di numeri casuali



```
097 numeri casuali/funzioni-casuali.c
  int rnd int(int min, int max) {
    return min + rand() % (max - min + 1);
3
4
  double rnd_double(double min, double max) {
    return min + (double)rand() / RAND_MAX * (max - min);
     man() {
. _rud_int(1,c) ...
```

$$r: \mathbb{N} \to \mathbb{N}$$

 $r_2 = f(r_1)$ $r_3 = f(r_2)$

 $r_n = f(r_{n-1})$

$$N \rightarrow N$$







seed



- La sequenza di numeri casuali restituiti da rand() in un programma è in realtà una sequenza di numeri pseudo-casuali calcolati ognuno dal precedente, a partire da un numero detto seed (seme).
- Se non impostato, il seed è 1. Quindi esecuzioni diverse di uno stesso programma generano la stessa sequenza di numeri casuali!
- Per impostare il seed, si può usare la funzione void <u>srand(unsigned int seed)</u>;. Ad esempio, l'istruzione <u>srand(100)</u>; impostà il seed a <u>100</u>.
- In questo modo però esecuzioni diverse dello stesso programma genereranno ancora sequenze uguali di numeri casuali (anche se diverse da quelle di seed 1).
- Per avere sequenze diverse, si può impostare il seed a un valore che varia necessariamente fra un'esecuzione e l'altra, ad esempio quello restituito dall'espressione time (NULL) (numero di secondi trascorsi dal 1 gennaio 1970).

Esempio



Simulazione di 10 lanci di un dado, con risultato che varia da esecuzione a esecuzione:

```
097 numeri casuali/dado-seed.c
   #include <stdio.h>
                                               Seed + 100
   #include <stdlib.h>
3
                                               rand() - 5000
   int main(void) { (MPOSTAZIONE SEME)
     int i;
     srand(time(NULL));
     for (i = 0; i < 10; i++)
       printf("%d\n", 1 + rand() % 6):
9
     return 0;
10
```