Generazione di numeri casuali in C++

La libreria C + + standard include un **generatore di numeri pseudo-casuali** per generare numeri casuali.
Per generare un numero casuale si usa la funzione **rand ()**.

```
1.  #include <iostream>
2.  using namespace std;
3.
4.  int main ()
5.  {int num = rand (); // genera un numero casuale
6.   cout << num << endl;
7.
8.   system("pause");
9.  }
```

La funzione **rand()** genera un numero che sarà compreso nell'intervallo da 0 a RAND_MAX, dove RAND_MAX è una costante definita a seconda del compilatore usato (nel caso nostro in ambiente Dev C++ è 32767).

Per scoprire il valore di RAND_MAX per il compilatore che utilizziamo scrivere il seguente codice C++:

```
1.  #include <iostream>
2.  using namespace std;
3.
4.  int main ()
5.  {cout << "Il valore di RAND_MAX e '" << RAND_MAX << endl;
6.   system("pause");
7.  }
```

Il generatore di numeri pseudo casuali produce una sequenza di numeri che danno l'impressione che sia casuale, in realtà la sequenza ad un certo punto si ripete e alla fine è prevedibile.
Proviamo a scrivere il seguente codice e dopo averlo compilato lo eseguiamo più volte:

```
1.  #include <iostream>
2.  using namespace std;
3.
4.  int main ()
5.  {cout <<rand()<<endl;
6.   cout <<rand()<<endl;
7.   cout <<rand()<<endl;
8.
9.   system("pause");
10. }
```

Ci accorgiamo che genera tre numeri pseudo-casuali ma, ad ogni esecuzione sono sempre gli stessi, perché la sequenza dei numeri viene prodotta a partire sempre dallo stesso valore iniziale (seme).
Per ovviare a questo inconveniente è necessario far generare la sequenza partendo ad ogni esecuzione con un valore diverso (un seme diverso).
Per questo occorre includere l'header **<time.h>** e richiamare prima della funzione rand() la funzione **srand()**.

```
1.  #include <iostream>
2.  #include <time.h>
3.  using namespace std;
4.
5.  int main ()
6.  {srand(time(NULL));      // Inizializza generatore di numeri pseudo-casuali
7.   cout <<rand()<<endl;
8.   cout <<rand()<<endl;
9.   cout <<rand()<<endl;
10.
11.  system("pause");
12. }
```

La funzione **srand ()** inizializza il generatore di numeri con un valore che passiamo come argomento (tra le parentesi tonde) nel nostro caso **time(NULL)**.

Ora, se eseguiamo più volte il programma, vediamo su video ogni volta tre numeri pseudo-casuali diversi dai precedenti.
Per comprendere meglio: la funzione **time(NULL)** restituisce un valore intero ottenuto dal clock interno, pari al numero di secondi trascorsi dal 1/1/1970 (se scriviamo l'istruzione cout<<time(NULL)<<endl; possiamo vedere su video il valore del tempo espresso in secondi dal 1/1/1970 all'istante in cui abbiamo fatto eseguire il programma).
In questo modo l'istruzione: **srand (time(NULL));** inizializza il generatore ad ogni esecuzione con un valore (seme) diverso e le successive istruzioni **rand()** produrranno valori differenti.
In genere abbiamo bisogno di chiamare **srand ()**, una sola volta, prima della prima chiamata la funzione **rand ()**.

Generazione di un numero in un intervallo specifico

Se vogliamo produrre numeri casuali in un intervallo specifico, piuttosto che tra 0 e RAND_MAX, possiamo usare l'operatore modulo (%).
Se usiamo: **rand ()% n** generiamo un numero da **0** a **n – 1**. Con l'aggiunta di un **offset** l'intervallo può essere modificato; ad esempio **rand()%10 + 1** produrrà numeri casuali da 1 a 10 anziché da 0 a 9.
Il seguente esempio genera 20 numeri pseudo-casuali nell'intervallo da 1 a 10:

```
1.  #include <iostream>
2.  #include <time.h>
3.  using namespace std;
4.
5.  int main ()
6.  {int x;
7.   srand(time(NULL));      // Inizializza generatore di numeri pseudo-casuali
8.   for ( int i = 0; i < 20; i++)
9.        {x = (rand() % 10) + 1;
10.  cout << x <<endl;
11. }
12.
13.  system("pause");
14. }
```

Caricamento di un vettore di 40 elementi con numeri pseudo-casuali compresi nel range da 20 a 100

Se vogliamo produrre numeri casuali da 20 a 100, dobbiamo usare l'operatore modulo (%) dividendo per la differenza tra il valore massimo (100) e il valore minimo (20) più 1, nel nostro caso 81=100-20+1. In questo modo **rand() % 81** genererà valori da 0 a 80. Sommando l'offset di 20 avremo valori da 20 a 100.

```
1.  #include <iostream>
2.  #include <time.h>
3.  using namespace std;
4.
5.  int main ()
6.  {int V[40];           // Dichiarazione del vettore
7.   srand(time(NULL));      // Inizializza generatore di numeri pseudo-casuali
8.   // Caricamento elementi del vettore
9.   for ( int i = 0; i < 40; i++)
10.       V[i] = (rand() % 81) + 20;
11.
12.  // Visualizza elementi vettore
13.  for ( int i = 0; i < 40; i++)
14.      cout << V[i] << "  ";
15.
16.  cout <<endl;
17.
18.  system("pause");
19. }
```