Random Numbers

SWINBURNE UNIVERSITY OF TECHNOLOGY

Random Numbers.

- Many engineering problems require the use of random numbers in the development of a solution.

- Random numbers that equally likely to occur in a specified range are called **uniform random numbers**.

- **`rand()`** function generates a random integer between 0 and RAND_MAX, inclusive (a common value for RAND_MAX is 32,767).

Random Numbers..

- Random == **completely** unpredictable, entirely "non-deterministic". "Impossible to predict next value from previous values".

- Program == a set of rules; **always** repeatable.
  - A true ‘Random Number’ program? Impossible!
  - C’s random numbers are not really random

- Instead: pseudo-random sequences (looks random)

Random Numbers…

Use the C function **`int rand(void);`**

□returns value is an integer between **0** and **`RAND_MAX`**

□Defined in **`stdlib.h`**

□Usage:

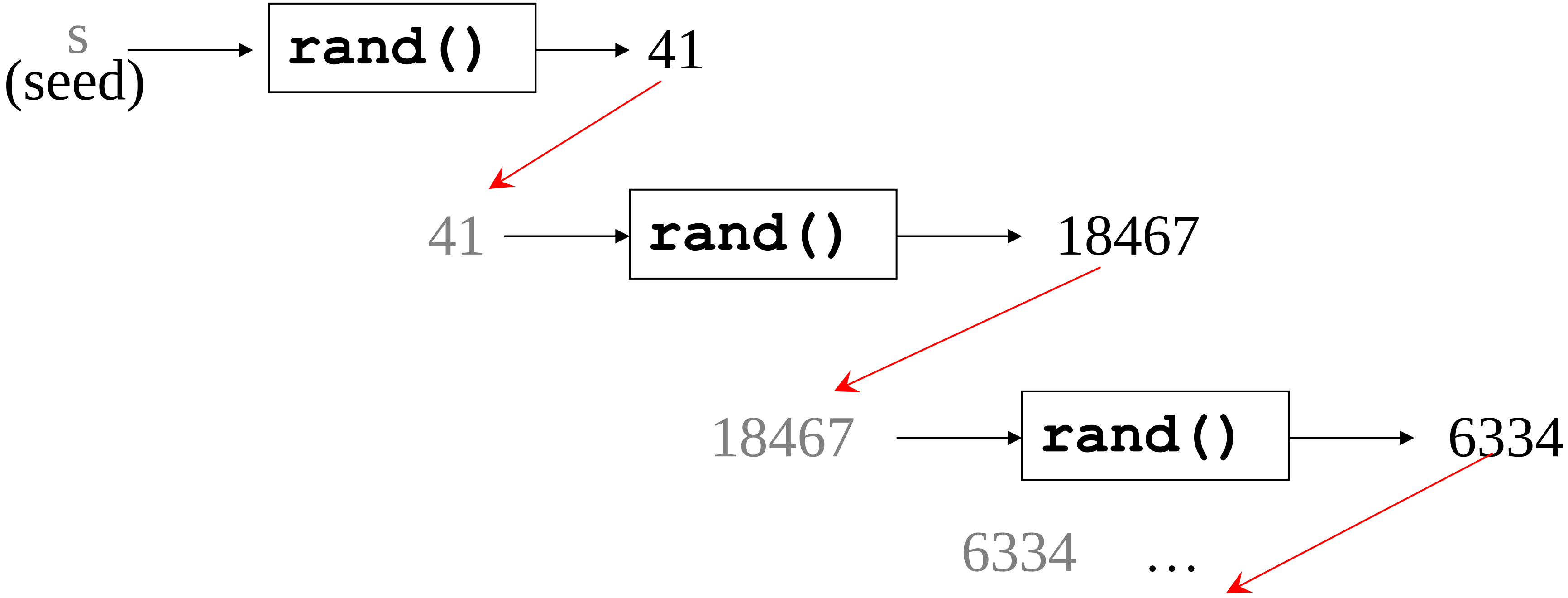
```
int myLuckyNumber = rand();
```

Pseudo-Random Number Sequence

How does it work?

- □ Result of previous **rand()** call is the (hidden) input to the bit-scrambling process that makes the next **rand()** answer.

- □ The initial hidden input value is called the **‘seed’**.



S
(seed)
`rand()`
41

41
`rand()`
18467

18467
`rand()`
6334

6334 …

Random Numbers….

- **REPEATABLE!** Same seed, same number of calls to **`rand()`** will give the same result every time.
- This is good for debugging:
  all problems are exactly repeatable.
- But usually bad for the appearance of randomness. Need to be able to SET the seed value…

Seed-Setting

- The initial value is called the ***seed***. Calls to `rand()` from the same seed give the same sequence.
- Seed-setting function is found in `stdlib.h:`
  - □ `void srand(unsigned seed);`
  - □ `usage: srand((unsigned)`*`seed`*`);`
- To guarantee the **same** pseudo-random sequence every time your program runs, set seed to a constant value (*e.g.* `srand(10);`)
- Use a **different** seed each time to guarantee a different pseudo-random sequence (but how?…)

Seed Setting..

Use the computer's time-of-day clock:

- □ Built-in C function to report current time found in `<time.h>` file; to access it use `#include <time.h>`
- □ The `time(NULL)` function call returns a time value, but is not an `unsigned` Must **cast** the returned value before use:

```
srand( (unsigned) time(NULL)); /* new rand seed */
```

- □ Careful! only need to call this ONCE in program; don't put it inside a loop. Try it as 1st statement of `main()`.

Random Numbers.....

- The default *random number seed* value is 1.

```
...
x=rand();
y=rand();
z=rand();
```

- Use `srand()` to change the seed . . .

```
scanf("%u", &seed);  /*unsigned integer*/
srand(seed);
x=rand();
y=rand();
```

- `srand((unsigned)seed);`

```
x=rand();
y=rand();
```

- **Program Chapter4_4** (Etter ~page 170)

Random Numbers

- To generate a random integer between 0 and 8:
  ```
  x = rand()%9;
  ```
- To generate a random integer between 2 and 78:
  ```
  x = rand()%77 + 2;
  ```
- To generate a random integer between -21 and 48:
  ```
  x = rand()%(48-(-21)+1)-21;
  ```
- To generate a random double value between 0 and 1:
  ```
  x = (double)rand()/RAND_MAX;
  ```
- To generate a random double value between a and b:
  ```
  x = ((double)rand()/RAND_MAX*(b-a))+a;
  ```

SWIN
BUR
NE

SWINBURNE
UNIVERSITY OF
TECHNOLOGY