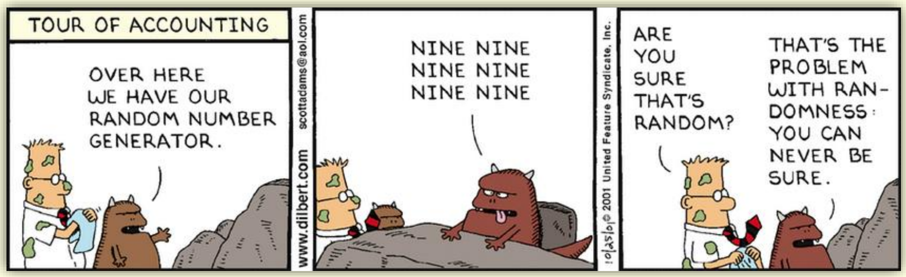
Random Numbers:
`rand()` *and* `srand()` *Library Functions*

CS 241
Data Organization using C

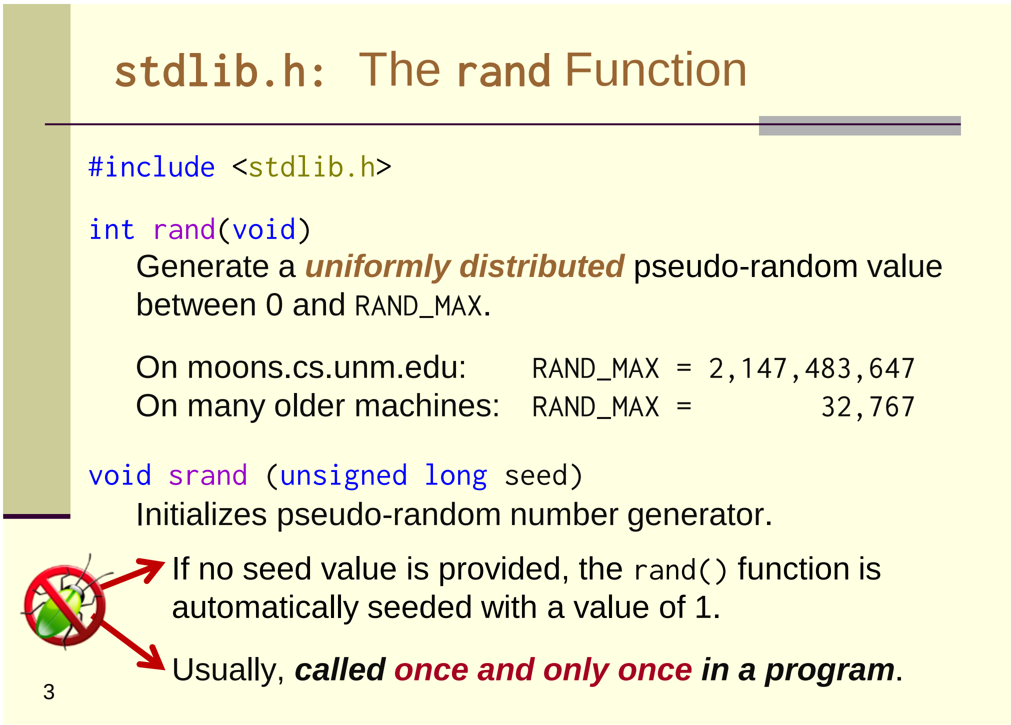
Instructor: **Joel Castellanos**
e-mail: joel@unm.edu
Web: http://cs.unm.edu/~joel/
Office: Farris Engineering Center
Room 2110

9/27/2019

DILBERT
by SCOTT ADAMS


TOUR OF ACCOUNTING
OVER HERE WE HAVE OUR RANDOM NUMBER GENERATOR.
scottadams@aol.com
www.dilbert.com
NINE NINE NINE NINE NINE NINE
10/25/01 © 2001 United Feature Syndicate, Inc.
ARE YOU SURE THAT'S RANDOM?
THAT'S THE PROBLEM WITH RANDOMNESS: YOU CAN NEVER BE SURE.

- What is **random**?
- What is **pseudorandom**
- Is the universe **deterministic**? If so, is random possible?
- If the universe is deterministic, is **freewill** possible?



stdlib.h: The rand Function

```
#include <stdlib.h>

int rand(void)
```

Generate a ***uniformly distributed*** pseudo-random value between 0 and `RAND_MAX`.

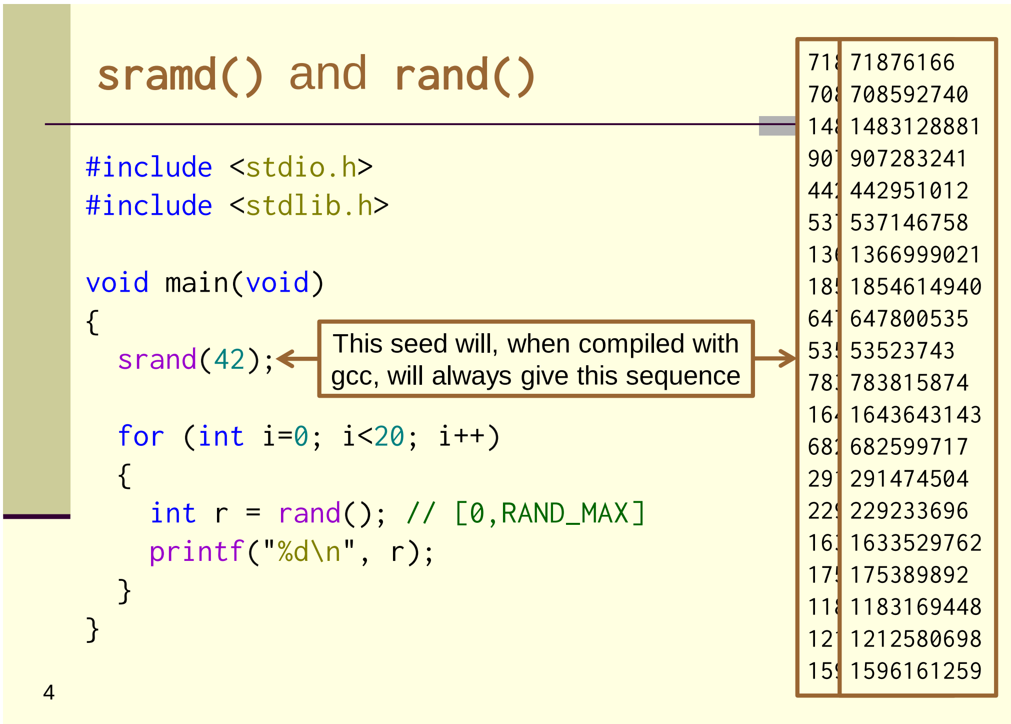
```
On moons.cs.unm.edu:       RAND_MAX = 2,147,483,647
On many older machines:    RAND_MAX =        32,767
```

```
void srand (unsigned long seed)
```

Initializes pseudo-random number generator.

If no seed value is provided, the `rand()` function is automatically seeded with a value of 1.

Usually, ***called once and only once in a program***.

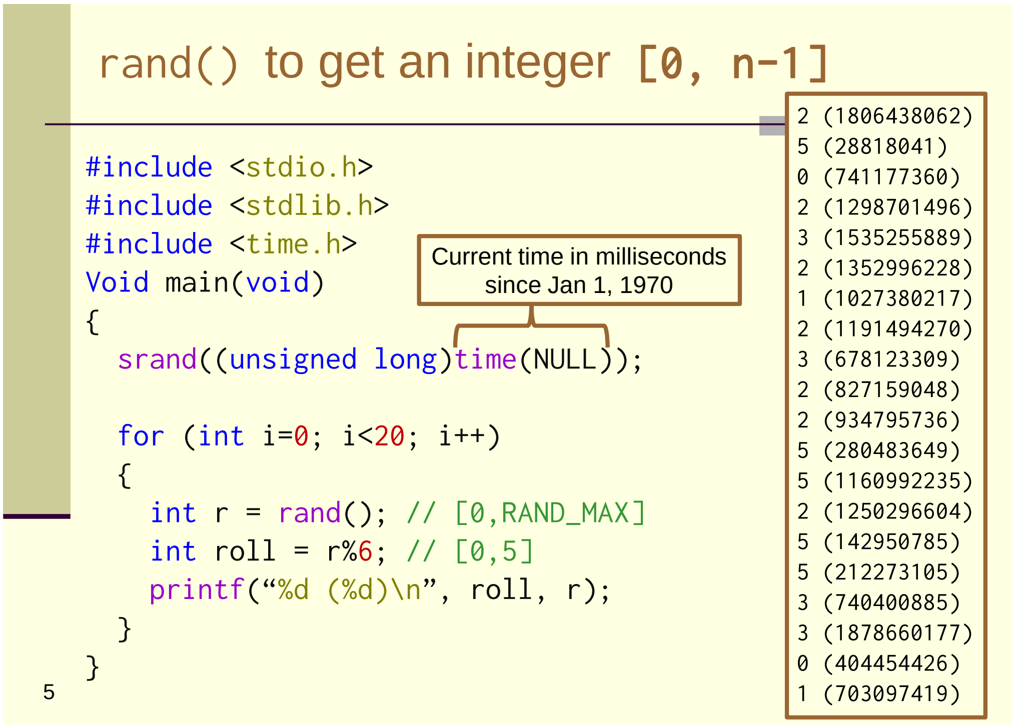


sramd() and rand()

```
#include <stdio.h>
#include <stdlib.h>

void main(void)
{
  srand(42);

  for (int i=0; i<20; i++)
  {
    int r = rand(); // [0,RAND_MAX]
    printf("%d\n", r);
  }
}
```

This seed will, when compiled with gcc, will always give this sequence

```
71 71876166
70 708592740
14 1483128881
90 907283241
44 442951012
53 537146758
13 1366999021
18 1854614940
64 647800535
53 53523743
78 783815874
16 1643643143
68 682599717
29 291474504
22 229233696
16 1633529762
17 175389892
11 1183169448
12 1212580698
15 1596161259
```

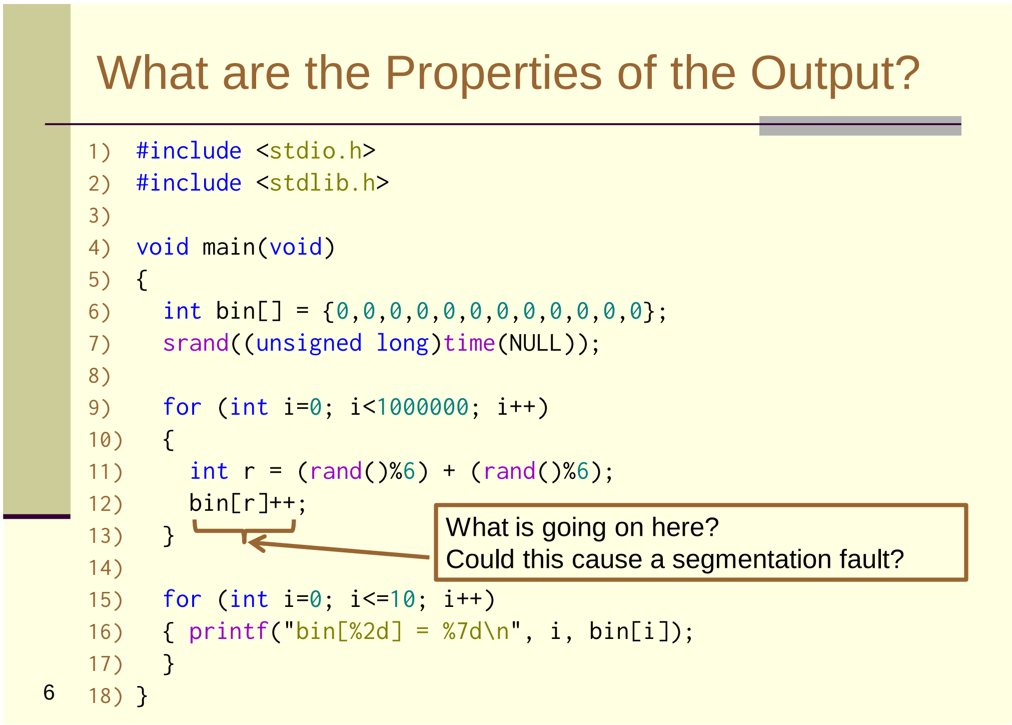


rand() to get an integer [0, n-1]

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
Void main(void)
{
  srand((unsigned long)time(NULL));

  for (int i=0; i<20; i++)
  {
    int r = rand(); // [0,RAND_MAX]
    int roll = r%6; // [0,5]
    printf(“%d (%d)\n”, roll, r);
  }
}
```

Current time in milliseconds since Jan 1, 1970

```
2 (1806438062)
5 (28818041)
0 (741177360)
2 (1298701496)
3 (1535255889)
2 (1352996228)
1 (1027380217)
2 (1191494270)
3 (678123309)
2 (827159048)
2 (934795736)
5 (280483649)
5 (1160992235)
2 (1250296604)
5 (142950785)
5 (212273105)
3 (740400885)
3 (1878660177)
0 (404454426)
1 (703097419)
```



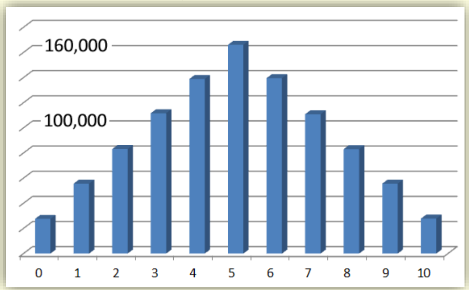
What are the Properties of the Output?

```
#include <stdio.h>
#include <stdlib.h>

void main(void)
{
  int bin[] = {0,0,0,0,0,0,0,0,0,0,0,0};
  srand((unsigned long)time(NULL));

  for (int i=0; i<1000000; i++)
  {
    int r = (rand()%6) + (rand()%6);
    bin[r]++;
  }

  for (int i=0; i<=10; i++)
  { printf("bin[%2d] = %7d\n", i, bin[i]);
  }
}
```

What is going on here?
Could this cause a segmentation fault?

Triangular Distribution

```
for (i=0; i<1000000; i++)
{
  int r = (rand()%6) + (rand()%6);
  bin[r]++;
}

for (i=0; i<=10; i++)
{ printf("bin[%2d] = %7d\n",
       i, bin[i]);
}
```



```
bin[ 0] =   27601
bin[ 1] =   55624
bin[ 2] =   83146
bin[ 3] =  111668
bin[ 4] =  138926
bin[ 5] =  166089
bin[ 6] =  139566
bin[ 7] =  110848
bin[ 8] =   83093
bin[ 9] =   55655
bin[10] =   27784
```

rand() getting a double [0.0, 1.0]

```
#include <stdio.h>
#include <stdlib.h>

double randomDouble()
{
  return (double)rand() / (double)RAND_MAX;
}

void main(void)
{
  srand((unsigned long)time(NULL));

  for (int i=0; i<20; i++)
  { printf("%f\n", randomDouble());
  }
}
```

```
0.167271
0.444177
0.854308
0.417682
0.966208
0.095919
0.043398
0.279063
0.789369
0.267750
0.621112
0.264053
0.401619
0.391749
0.969598
0.581943
0.379239
0.883799
0.872260
0.344283
```