Function Call Stack and Activation Frame

- Stack
  - Just like a pile of dishes
  - Support Two operations
    - push()
    - pop()
  - LIFO (Last-In, First-Out) data structure
    - The last item pushed (inserted) on the stack is the first item popped (removed) from the stack

Function Call Stack

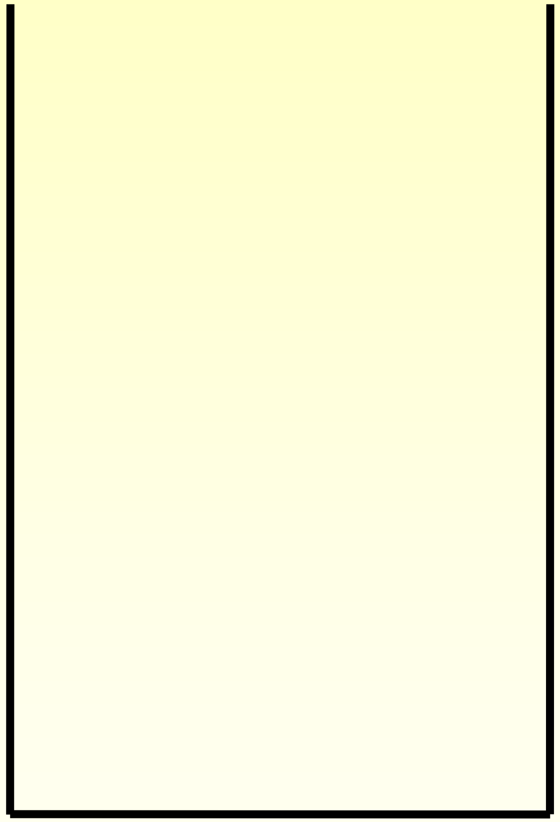
- Supports the function call/return mechanism
  - Each time a **function calls** another function, a stack frame (also known as an **activation record**) is pushed onto the stack
    - Maintains **the return address** that the called function needs to return to the calling function
    - Contains automatic variables—**parameters and any local variables** the function declares
  - When the called **function returns**
    - **Stack frame** for the function call is **popped**
    - **Control transfers to the return address** in the popped stack frame
- Stack overflow
  - Error that occurs when more function calls occur than can have their activation records stored on the function call stack (due to memory limitations)

Function call stack

```
int fcn1 (int local_int)
{
    int x;
    x = fcn2(local_int * 2);
    return (x - 3);
}

int fcn2 (int local_int2)
{
    int y = local_int2 - 85;
    return y;
}

int main()
{
    int z;
    z = fcn1 (10);
    printf("%d\n",z);
    return 0;
}
```



Random function

- rand()   function
  - Declared in `<stdlib.h>`
  - generate a random (integer) number between 0 and '`RAND_MAX`' (at least 32767).
  - `RAND_MAX` is defined in a header file (`stdlib.h`)
  - srand is necessary for selecting a “SEED” for random number generation
  - Syntax

```
int r;

r = rand();
```

srand() function

- **`srand`** seeds the random number generation function `rand()` so it does not produce the same sequence of numbers

- Library: `stdlib.h`
- Prototype: void `srand(unsigned int seed);`
- Syntax:

```
unsigned int seed=10; /* seed value */
srand(seed);
```

- Quite often, we use

```
srand(time(NULL)); // seed value will be current time
```

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

int main()
{
        int count1=0, count2=0, count3=0, count4=0, count5=0, count6=0;
        int face, roll;

        srand(time(NULL));

        for (roll=0 ; roll <= 6000 ; roll++) {
                face = 1 + rand() % 6 ;     // get a random number between 1 and 6
                switch (face) {
                        case 1 :
                                count1++;
                                break;
                        case 2 :
                                count2++;
                                break;
                        case 3 :
                                count3++;
                                break;
                        case 4 :
                                count4++;
                                break;
                        case 5 :
                                count5++;
                                break;
                        case 6 :
                                count6++;
                                break;
                        default :
                                printf("This case is impossible!\n");
                                break;
                }
        }
printf("1 : %5d\n",count1);
        printf("2 : %5d\n",count2);
        printf("3 : %5d\n",count3);
        printf("4 : %5d\n",count4);
        printf("5 : %5d\n",count5);
        printf("6 : %5d\n",count6);

        return 0;
}
```

output

```
1 :  1050
2 :  1033
3 :   964
4 :   983
5 :   981
6 :   990
```

Using Array?