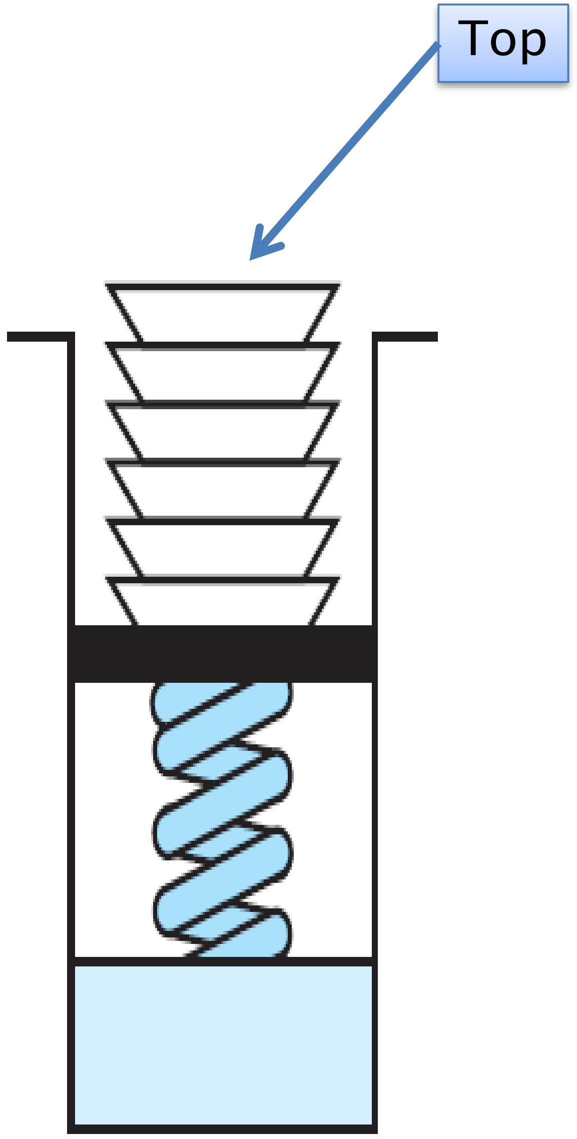
Recursion

And Stack Frames

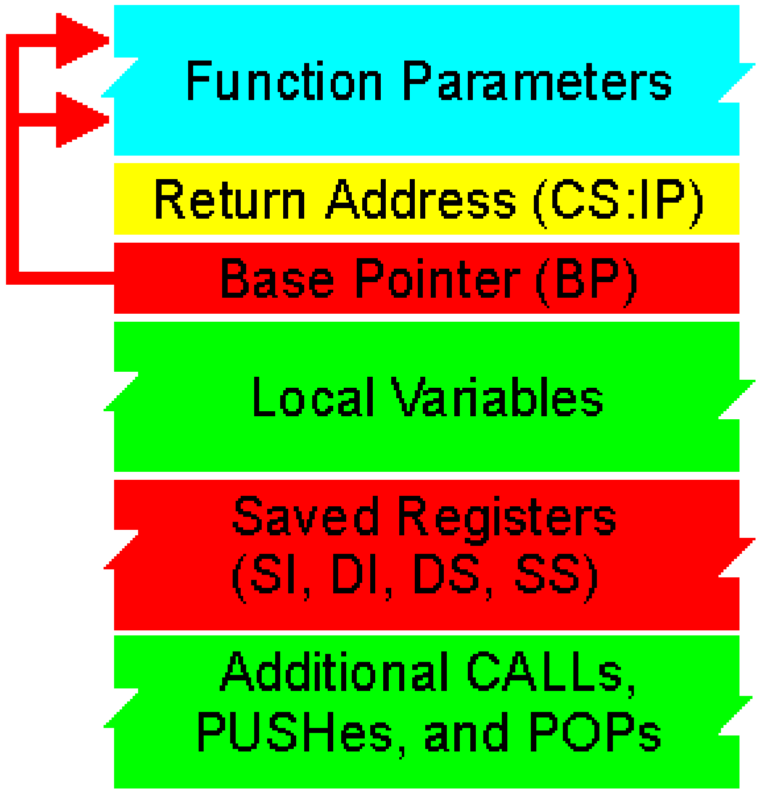
ADT Stack

- A Stack is an abstract data type where all operations are performed at the ‘top’ of the Stack, similar to a pile of plates in the cafeteria.

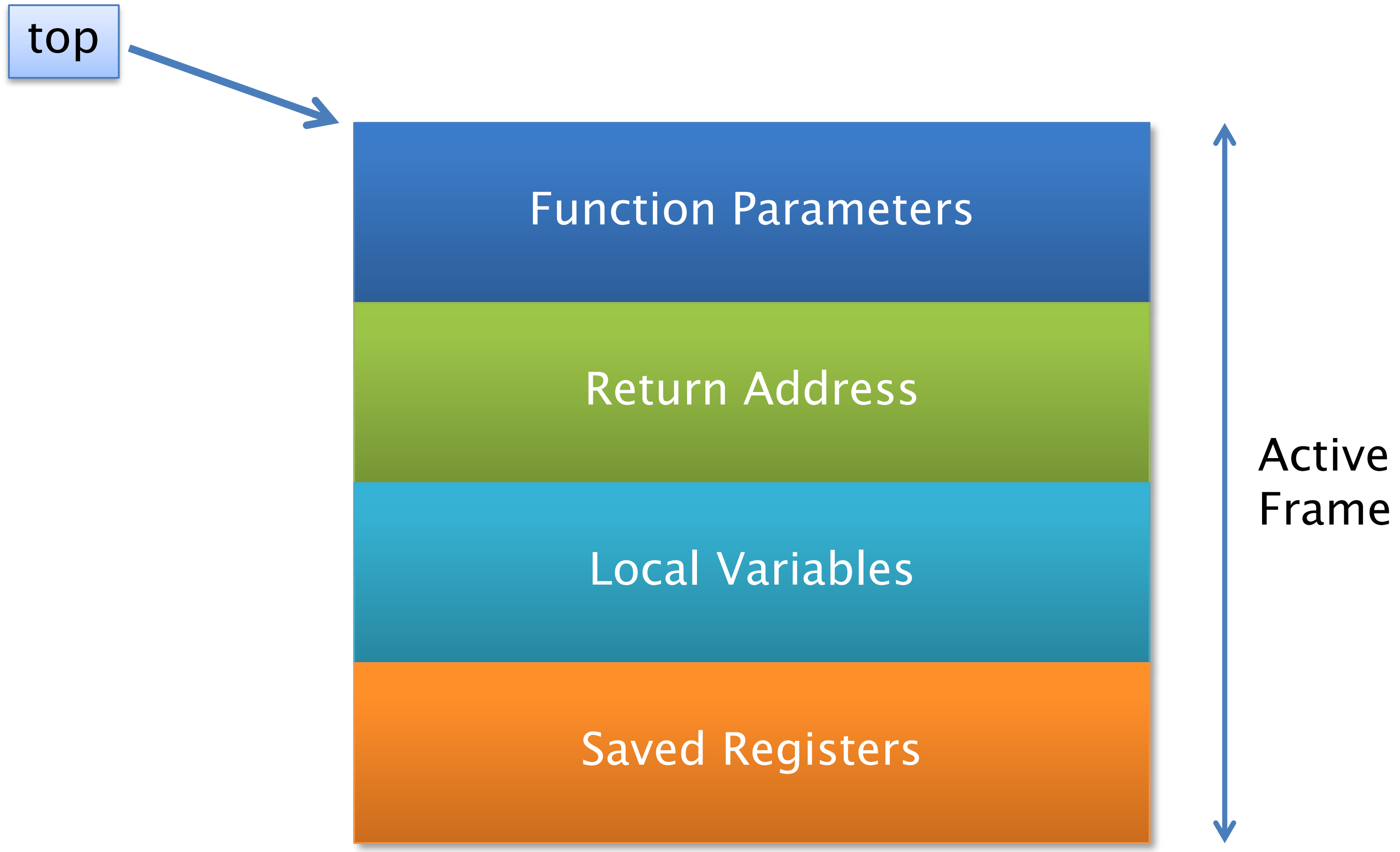


Stack Frame

- A Stack frame is produced by the C++ compiler every time a function is called.



Stack Frame Diagram



Stack Frame: What is it?

- A structure used to save a function’s data.
- The frames are ‘pushed’ when calling a function.
- The frames are ‘popped’ when returning from a function.

Example Program

```
int function(string& str)
{
      string s = "[a-zA-Z]";
      str = s;
      return s.length;
}
void main()
{
      string astring;
      function( astring );
}
```

Function parameters

Local variables

Function call

Calling the Function from Main

- Main() saves the local and temporary variables by pushing them onto the stack.
- The parameters to function() are pushed onto the stack.
- The ‘call’ instruction is executed to transfer control to function().

Inside Function

- The current instruction pointer is pushed on the stack.
- The frame pointer is updated with the current stack pointer.
- The local variables inside the function are accessed.

Returning from the Function

- Replaces the stack pointer with the current frame pointer.
- Pops the instruction pointer.
- Returns to main() by popping the stack frame.

Recursion Example

```
void Bitshow(int number)
{
        if (number)
        {
                Bitshow( number>>1 );
                if (number%2)
                        cout << "1";
                else
                        cout <<"0";
        }
}
```

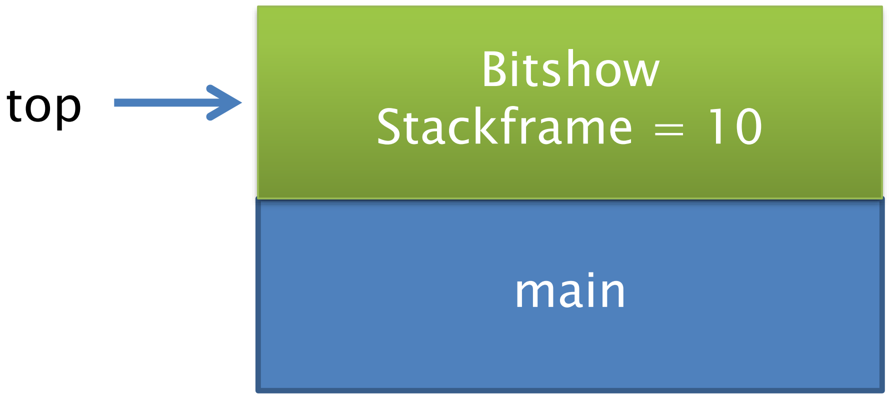
Recursion

Calling the Recursive Function

- Initial call:

```
int number = 10;  // bit values 1010
bitshow(number);
```

This puts a stack frame with 10 as the parameter on the Function Argument part of the frame.

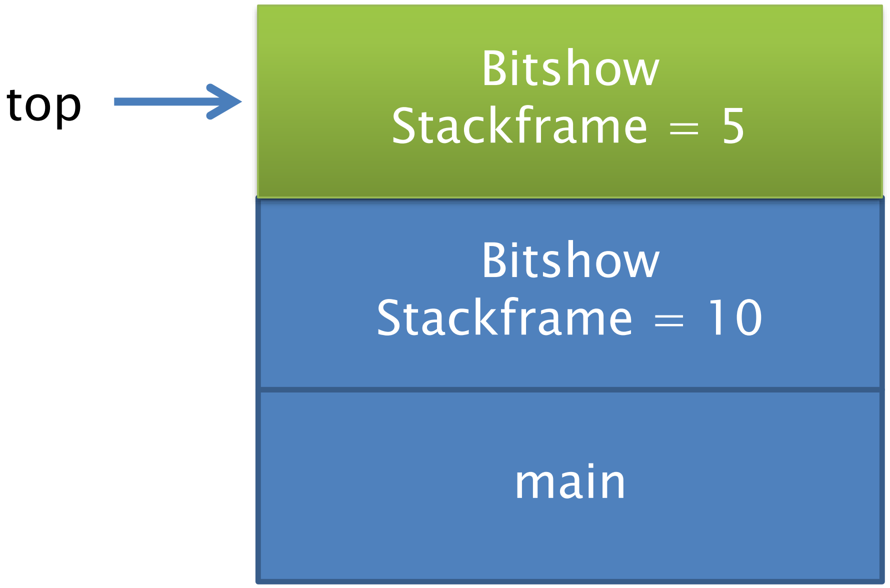

top
Bitshow
Stackframe = 10
main

Inside Bitshow

- Since 10 is non-zero, the if statement is entered, and the recursive call to bitshow is hit:

```
bitshow( number >> 1 );
```

Ten shifted one is 10 / 2 which equals 5. Bitshow is recursively called again, and a stackframe is created with 5 for the function parameter.

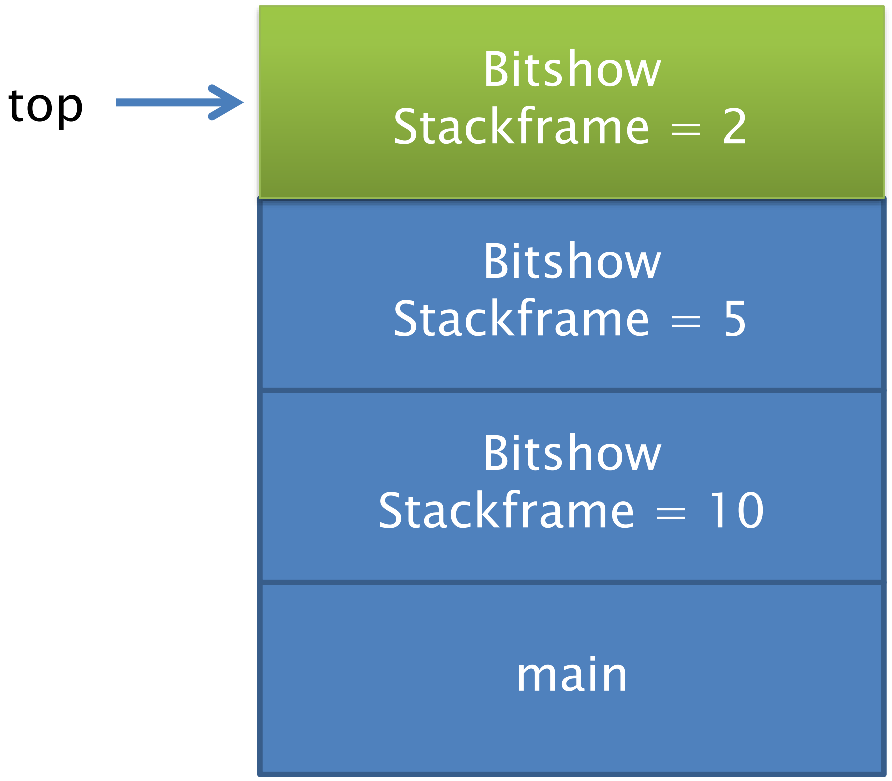

top
Bitshow
Stackframe = 5
Bitshow
Stackframe = 10
main

More Bitshow

- Since 5 is non-zero, the if statement is entered, and the recursive call to bitshow is hit:

```
bitshow( number >> 1);
```

  5 shifted one is 5 / 2 which equals 2. Bitshow is recursively called again, and a stackframe is created with 2 for the function parameter.

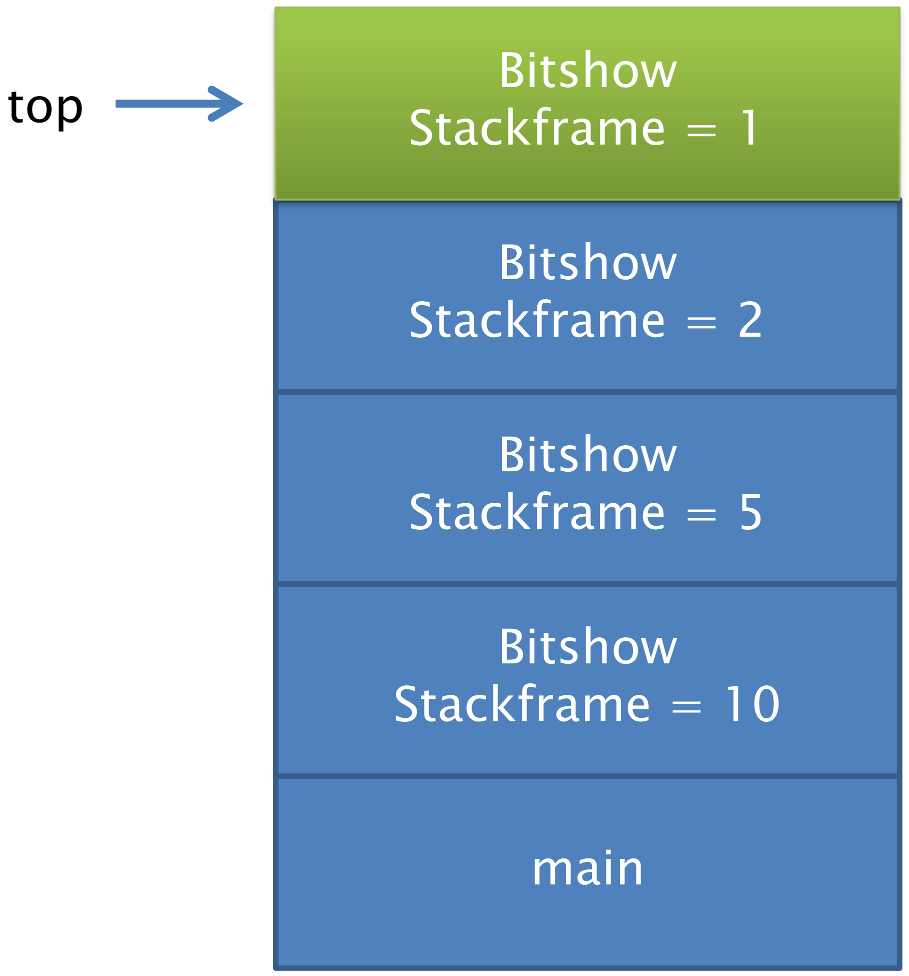

top
Bitshow Stackframe = 2
Bitshow Stackframe = 5
Bitshow Stackframe = 10
main

More Bitshow

- Since 2 is non-zero, the if statement is entered, and the recursive call to bitshow is hit:

```
bitshow( number >> 1 );
```

  2 shifted one is 2 / 2 which equals 1. Bitshow is recursively called again, and a stackframe is created with 1 for the function parameter.


top
Bitshow Stackframe = 1
Bitshow Stackframe = 2
Bitshow Stackframe = 5
Bitshow Stackframe = 10
main

More Bitshow

- Since 1 is non-zero, the if statement is entered, and the recursive call to bitshow is hit:

```
    bitshow( number >> 1);
```

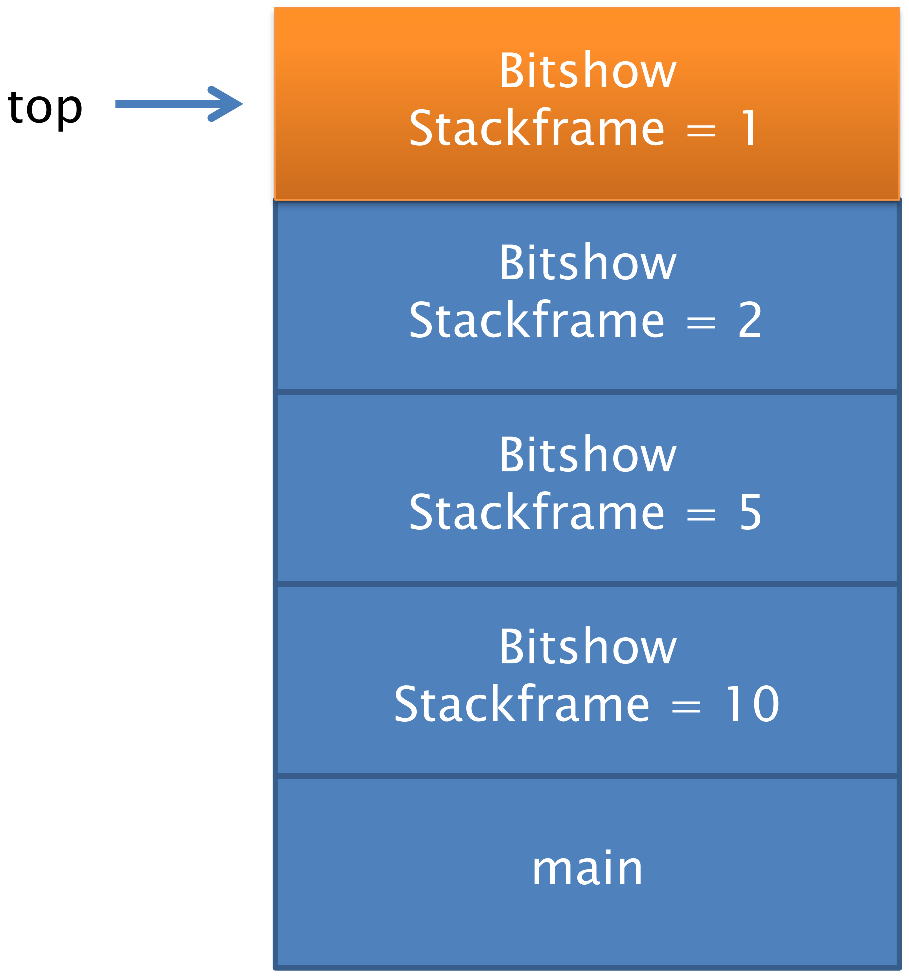
  1 shifted one is 1 / 2 which equals 0. Bitshow is recursively called again, and a stackframe is created with 0 for the function parameter.

End Bitshow Recursion

- Since 0 is false, the if statement is not entered, and the recursive call to bitshow is not hit.
- Now it's time to pop the stack frames, with the statement following the recursive calls:

```
    if (number % 2)
```

Stackframe Pop



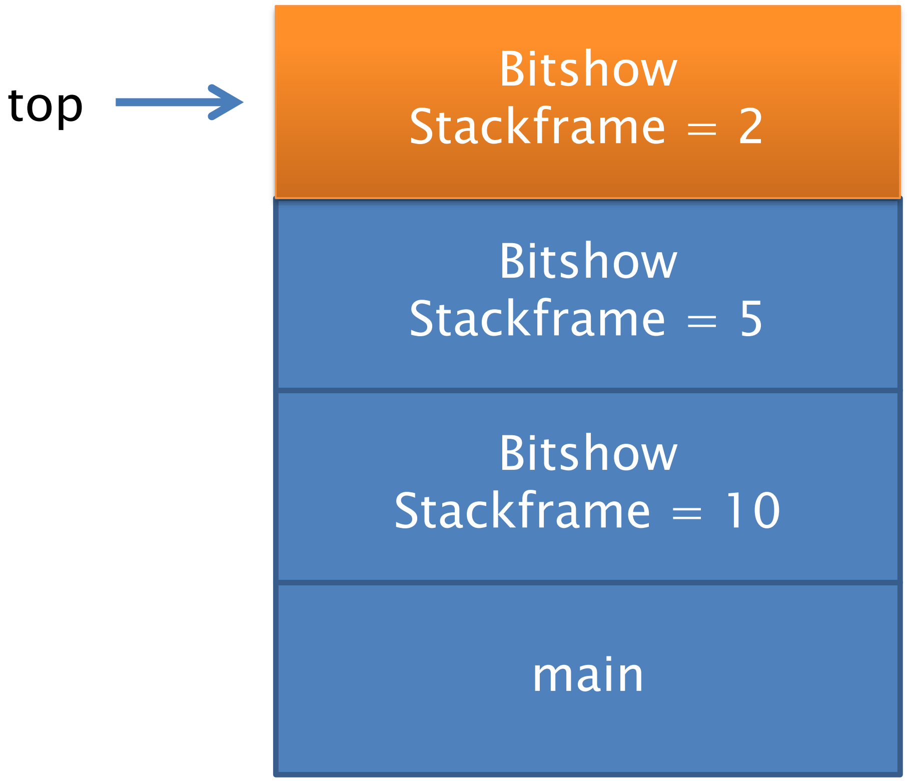
```
if (number % 2)
        cout << "1";
else
        cout << "0";
```

Bitshow
Stackframe = 1

Since 1 % 2 = 1, a 1 is printed

Result: 1

Stackframe Pop



Stackframe Pop

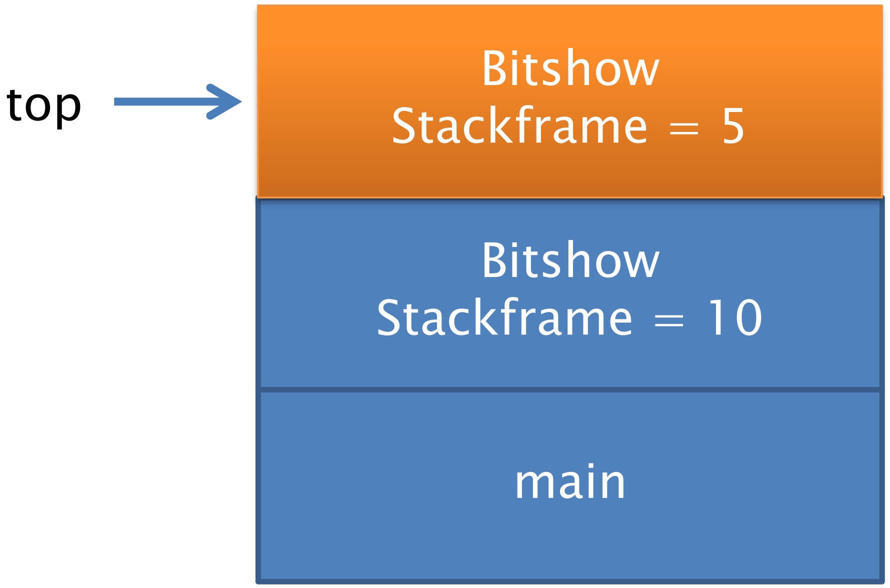
```
if (number % 2)
        cout << “1”;
else
        cout << “0”;
```

Bitshow
Stackframe = 2

Since 2 % 2 = 0, a 0 is printed

Result: **10**

Stackframe Pop



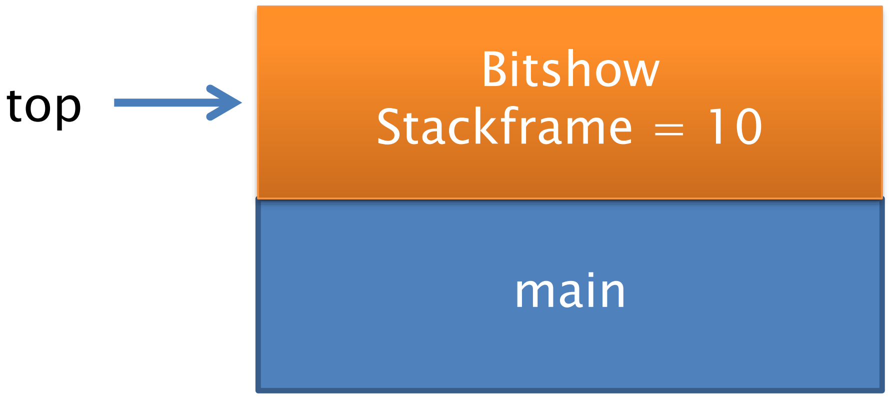
```
if (number % 2)
        cout << “1”;
else
        cout << “0”;
```

Bitshow
Stackframe = 5

Since 5 % 2 = 1, a 1 is printed

Result: **101**

Stackframe Pop



```
if (number % 2)
        cout << "1";
else
        cout << "0";
```

Bitshow
Stackframe = 10

Since 10 % 2 = 0, a 0 is printed

Result: **1010**

Back to the Main Stackframe

The last thing on the Stackframe stack is the stackframe for main: