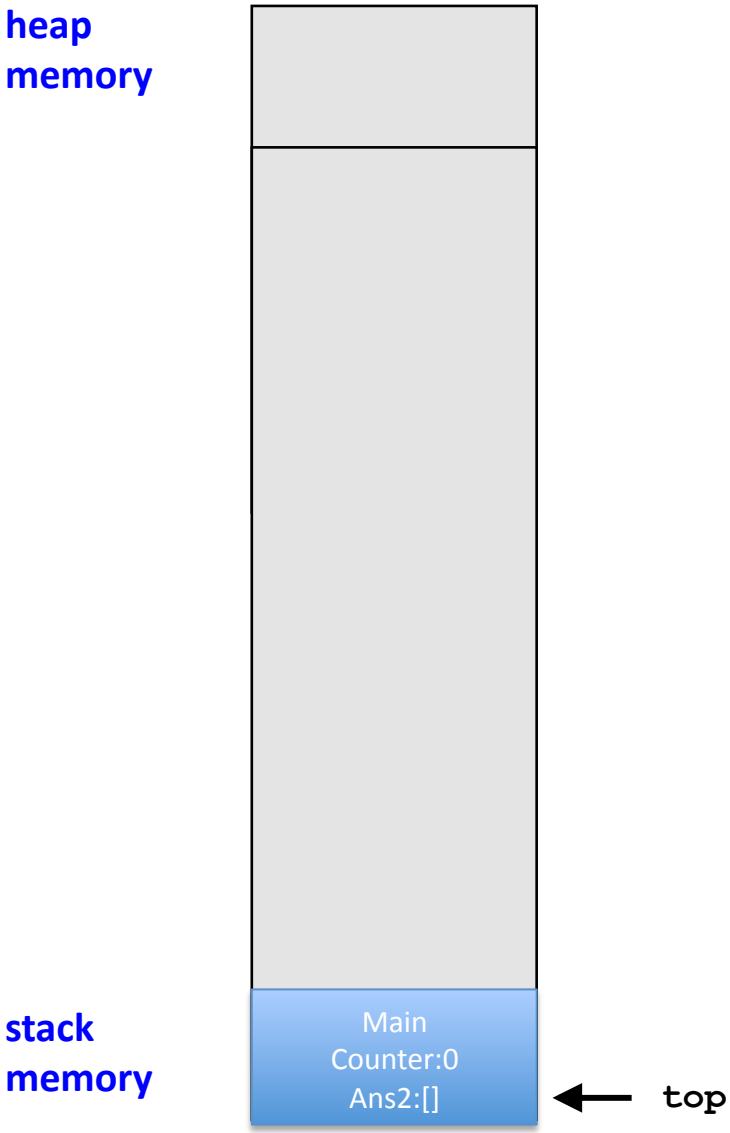
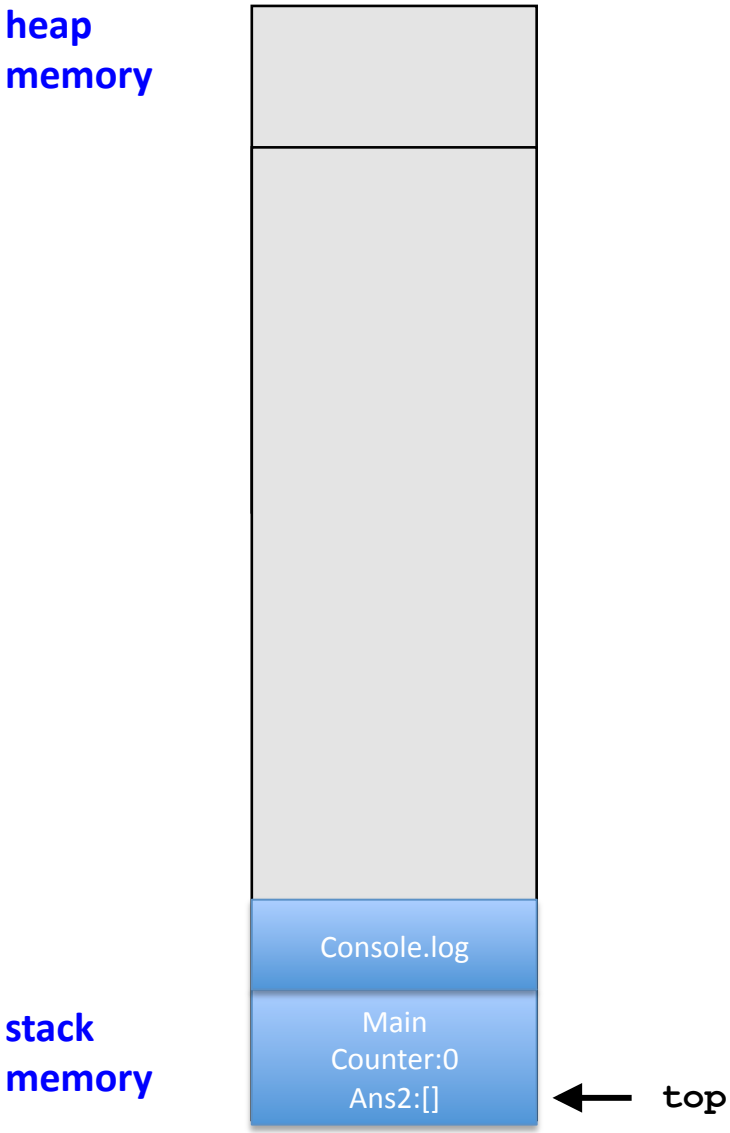
# Local Scope

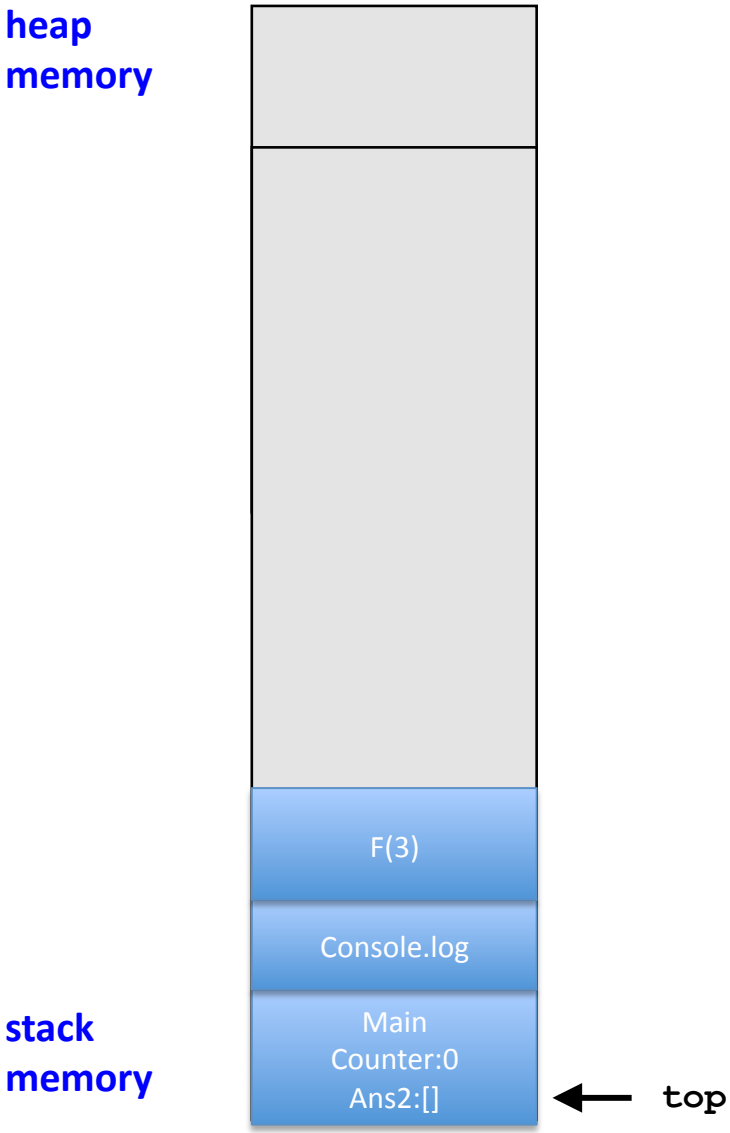
R2.2/2.3



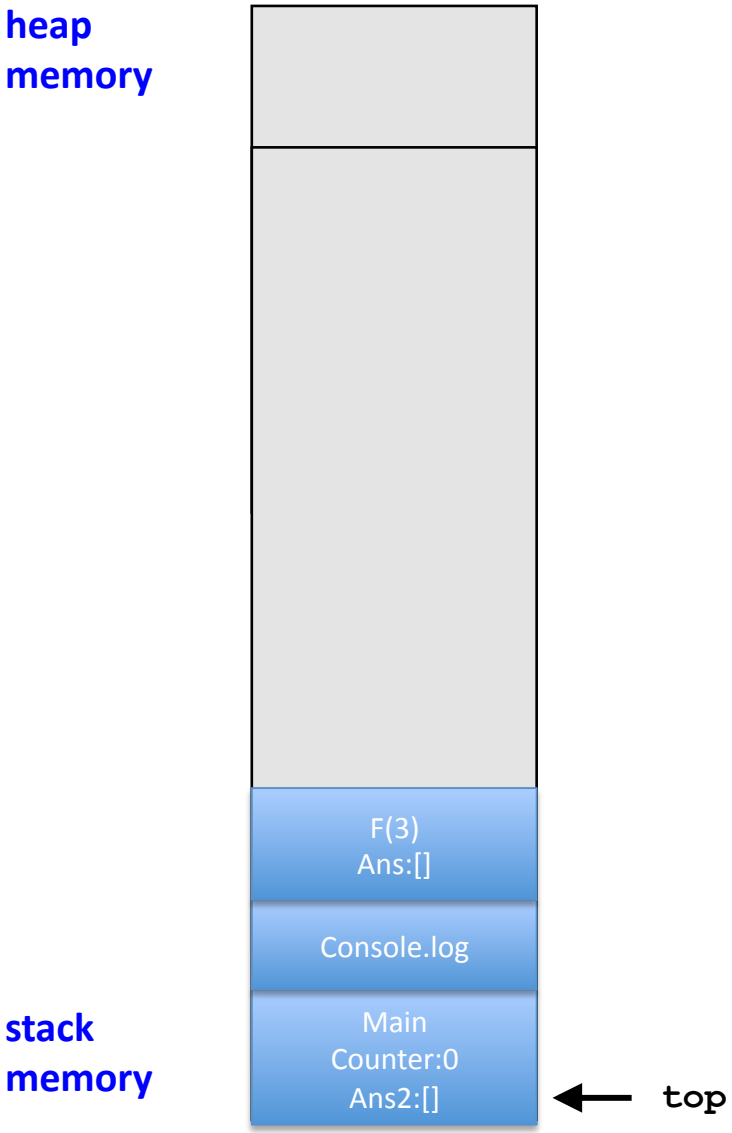

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable fo f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



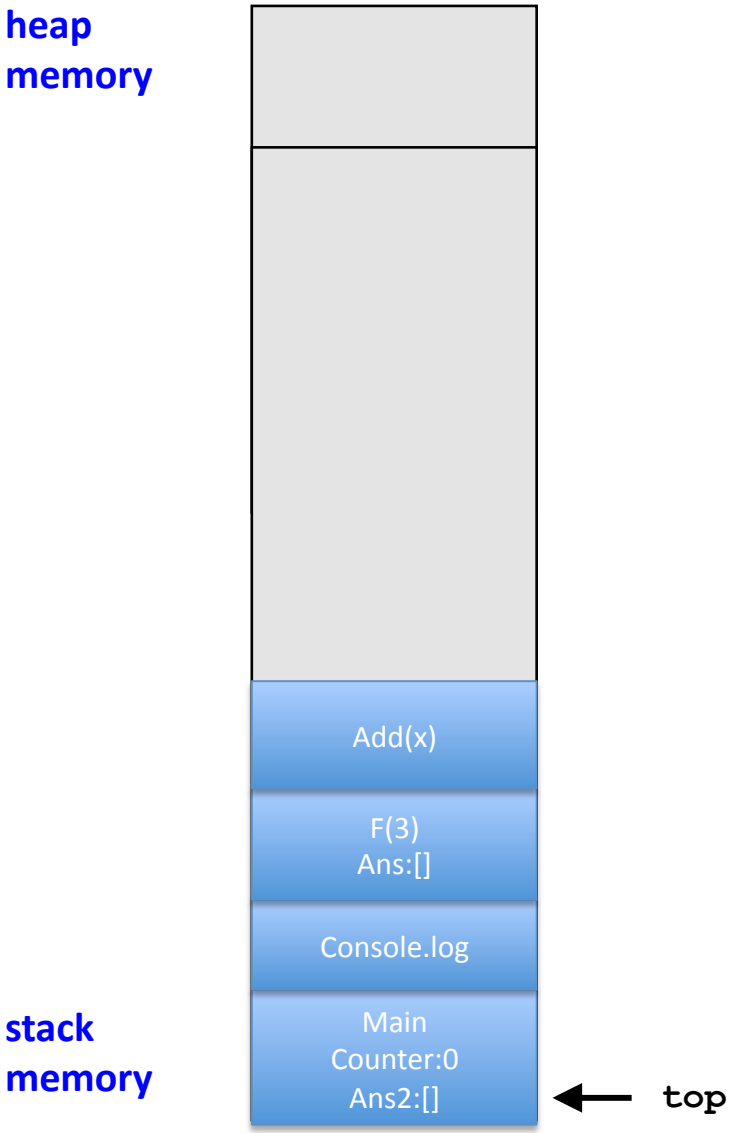

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable fo f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



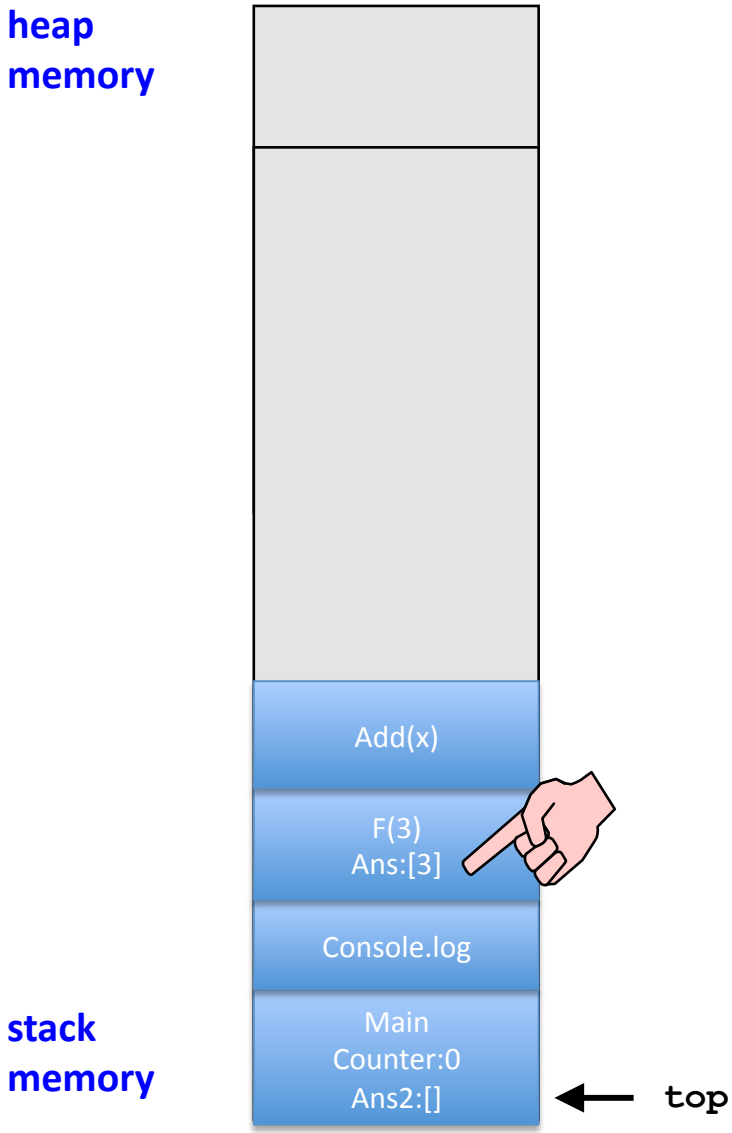

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable fo f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}


console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



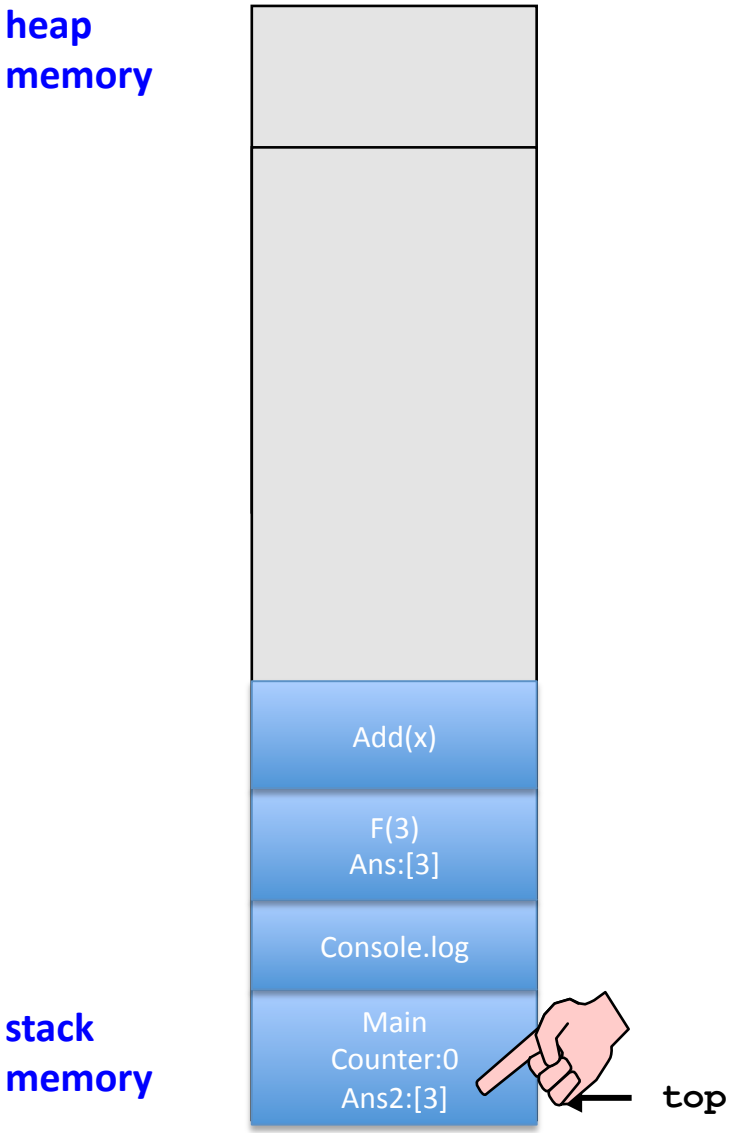

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable fo f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



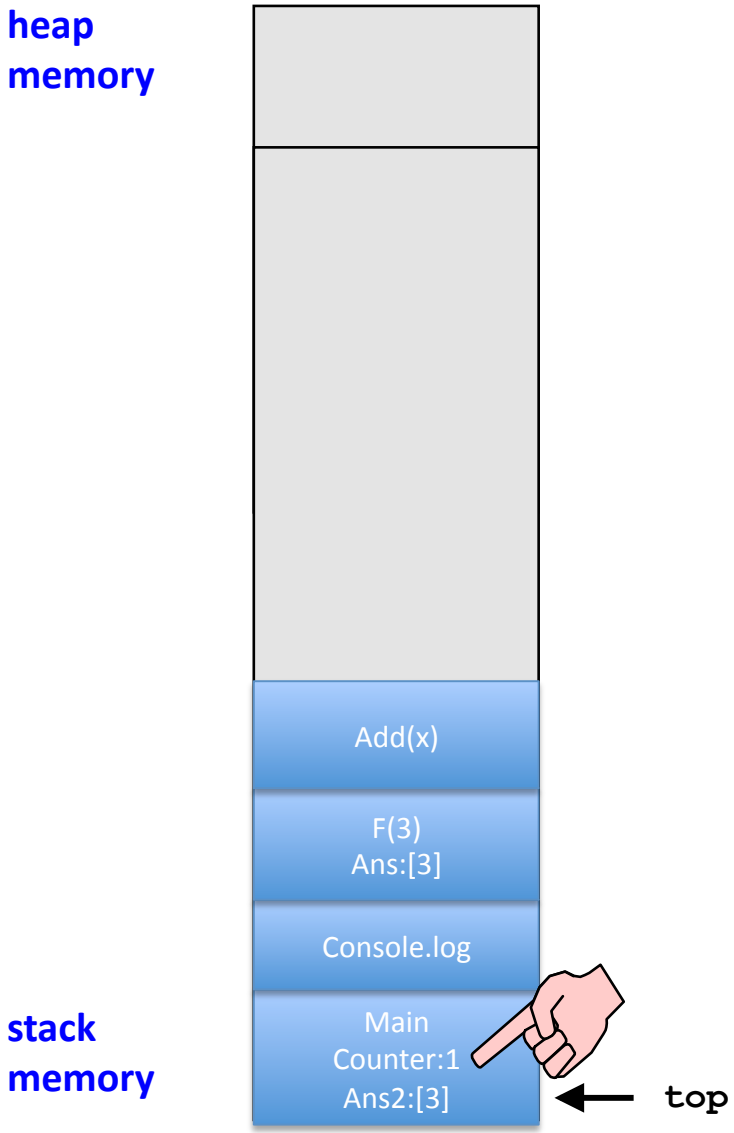

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable fo f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



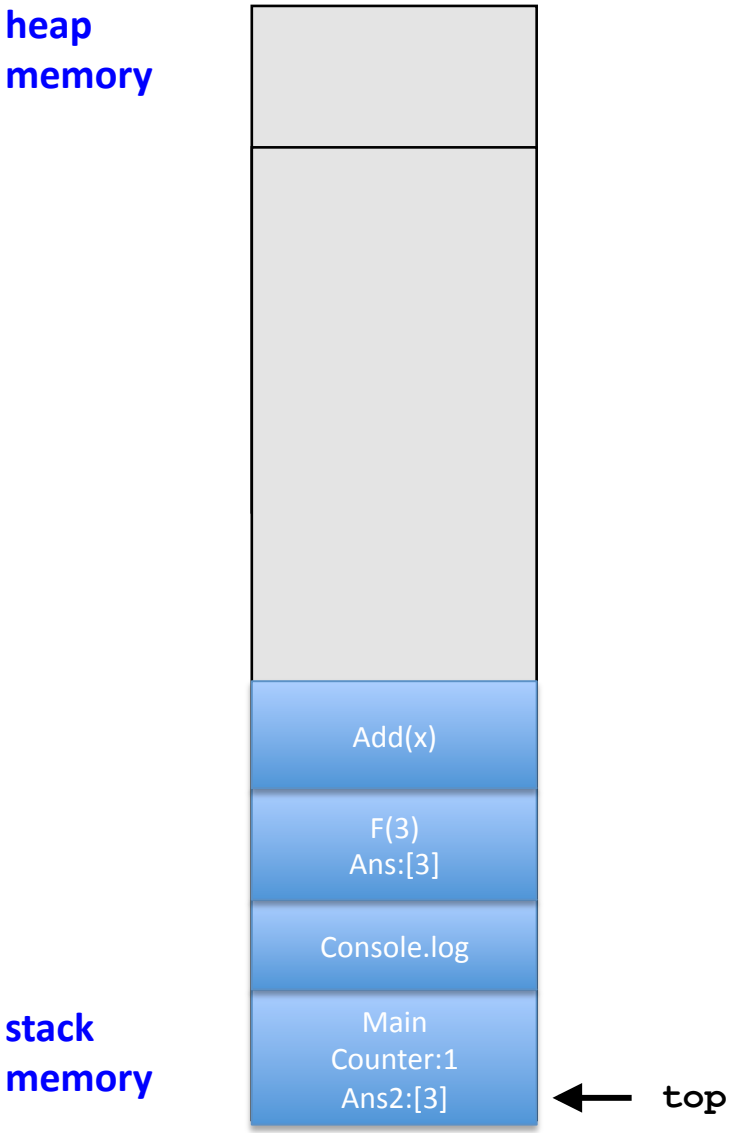

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local [illegible]ble of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable for f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



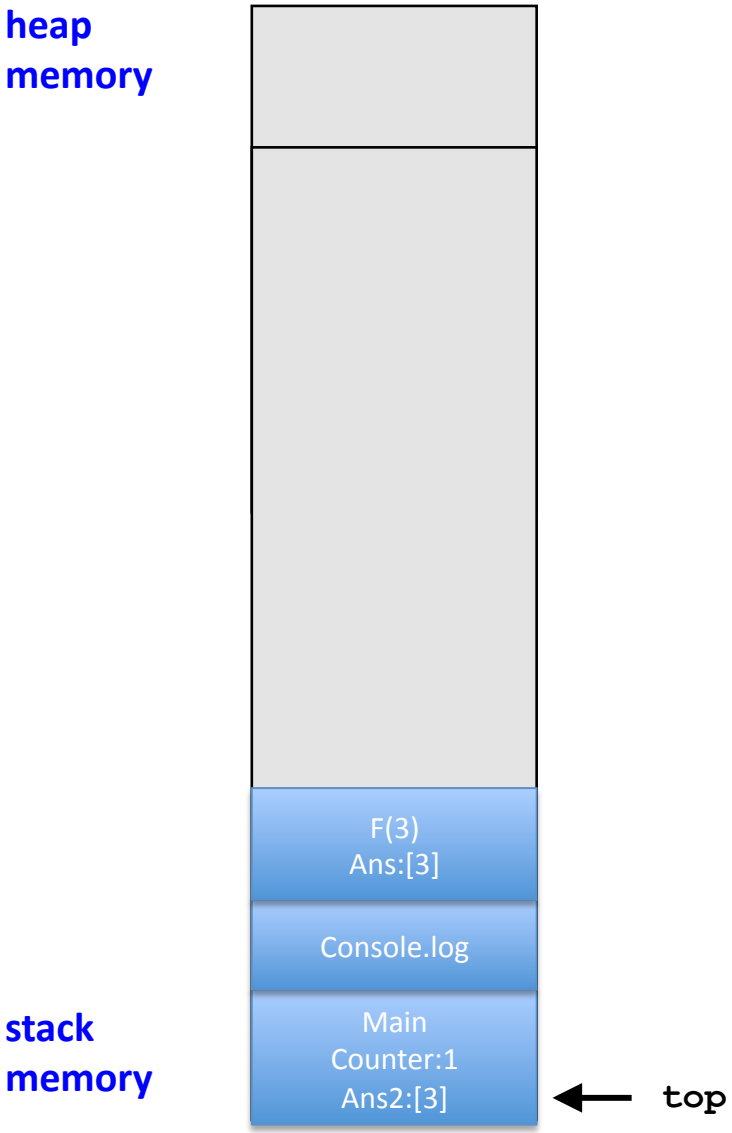

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable for f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



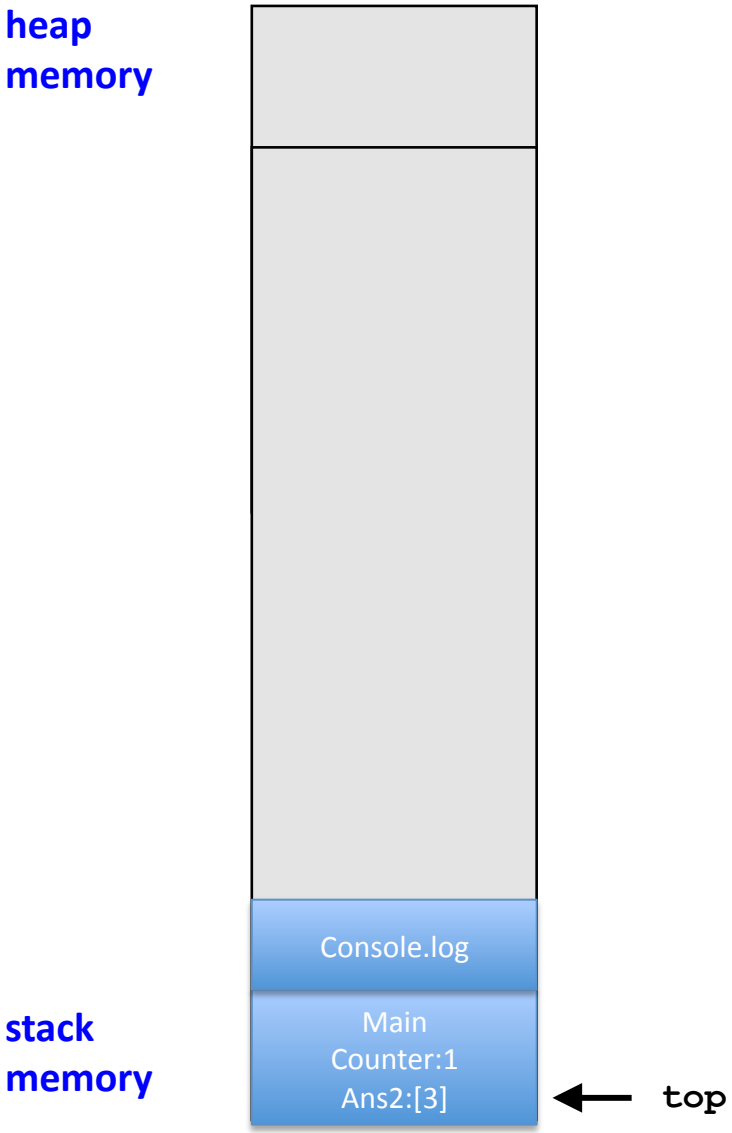

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable for f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



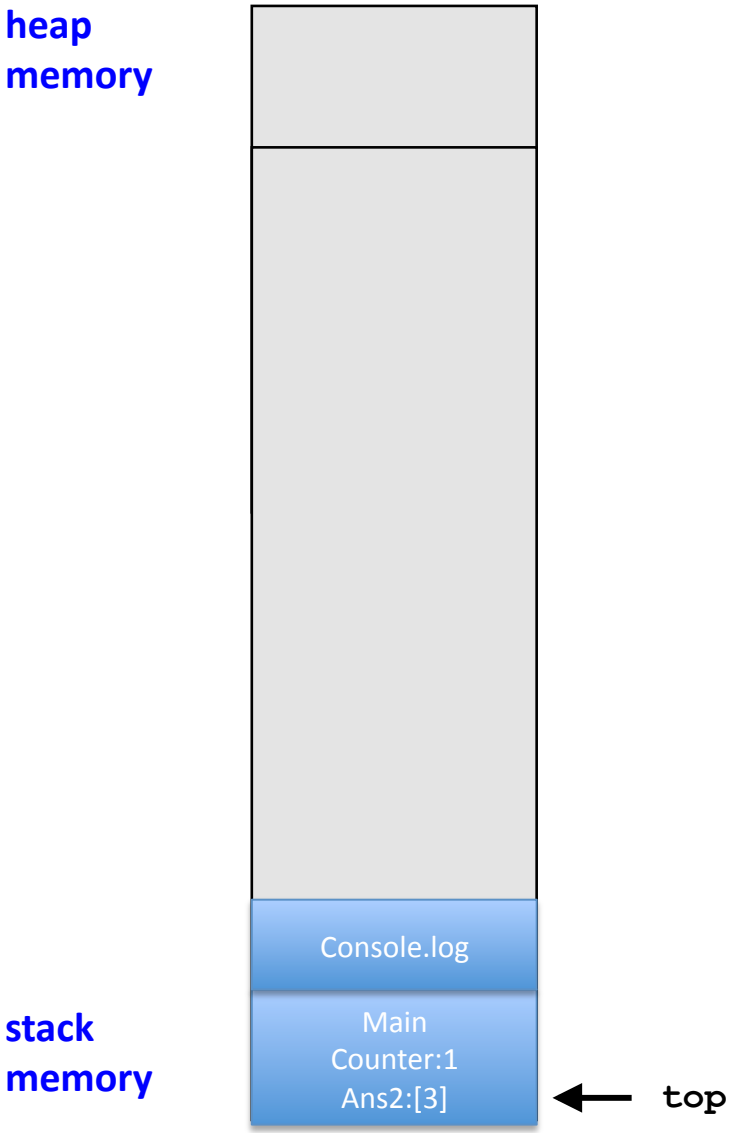

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable for f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```



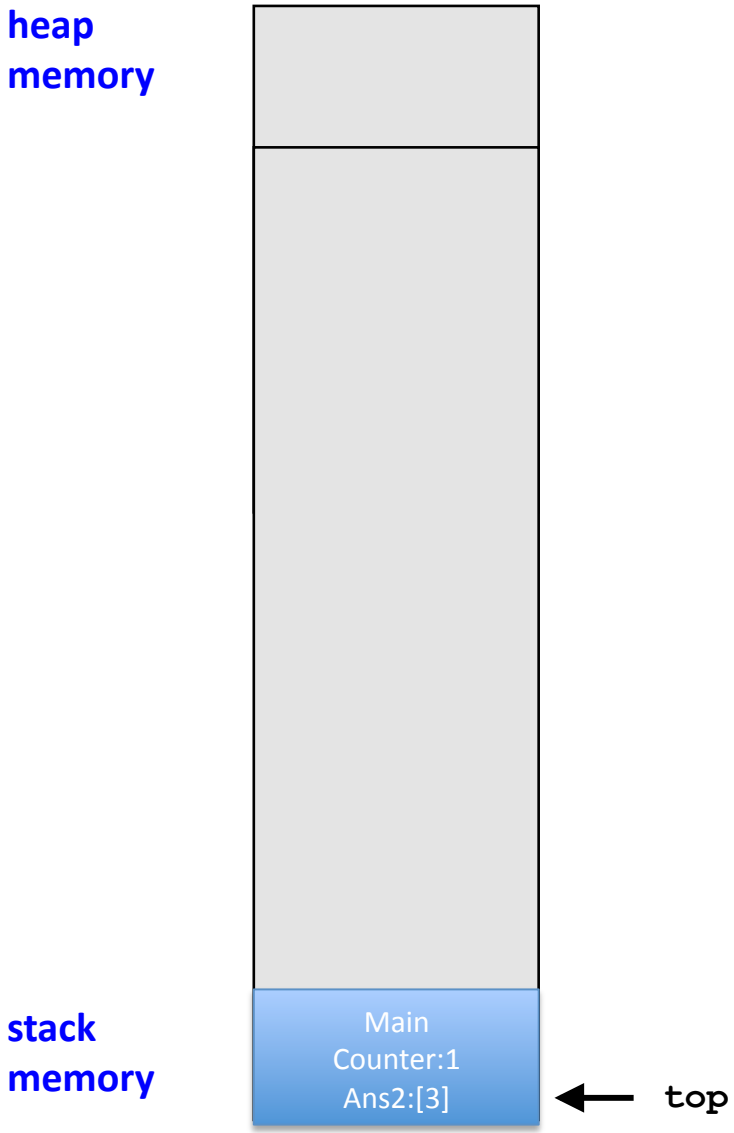

```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable for f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```




```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable for f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```




```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable for f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```




```
//this code simply tests the effects of local
     scope
var counter=0;

var ans2=[];//global variable

function f(x){

var ans=[];//local variable of f(x)

function add(x){

     ans[counter]=x;//accessing the local
     variable for f(x)

     ans2[counter]=x;//accessing the global
     variable

     counter++;

     return ans;

}

return add(x);

}



console.log(f(3));//log the example

console.log(ans2);//log the global variable
```

# Conclusion/R2.3

- No modifications are required for my example to work, it's a fairly simple one.
- I would suggest a better way to keep track of a variable through a "reference" notation, so that we only need to change the base case.