C Programming Language

(4th class)

Dohyung Kim

Assistant Professor @ Department of Computer Science

Today …

- **Function**

Function Example

```
#include <stdio.h>
int main (void)
{
    int i;
    for (i=0; i<5; i++) {
        printf(“the value of i : %d\n”, i*i );
    }
    for (i=0; i<5; i++) {
        printf(“the value of i : %d\n”, i*i );
    }
    return 0;
}
```

```
#include <stdio.h>
int PrintSquares(int val)
{
    int i, res=0;
    for (i=0; i<val; i++) {
        printf(“the value of i : %d\n”, i*i );
    }
}

int main (void)
{
     PrintSqures(5);
     PrintSqures(5);
     return 0;
}
```

Function Example

```
#include <stdio.h>
int Squares(int x)
{
    int sq_x;
    sq_x = x * x;
    return sq_x;
}

int main (void)
{
    int res = 0;
    res = Square(3);
    printf(“the square of 3 : %d \n”, res);
    printf(“the square of 5 : %d \n”, Squres(5));
    return 0;
}
```

Terminology

- In the previous examples
  - The Main function **calls** PrintSquares and Squares.
  - A function's input values are known as it **arguments** (**parameters**).
  - A function that gives some kind of answer back to the caller (function) is said to **return** the answer.

Define Function

```
type name (type1 arg1, type2 arg2, … )
{
    /* code */
}
```

```
int Squares(int x)
{
    int sq_x;
    sq_x = x * x;
    return sq_x;
}

void print_result (int a, int b)
{
    printf(“the result is : %d, %d \n”, a, b);
}
```

Declare A Function

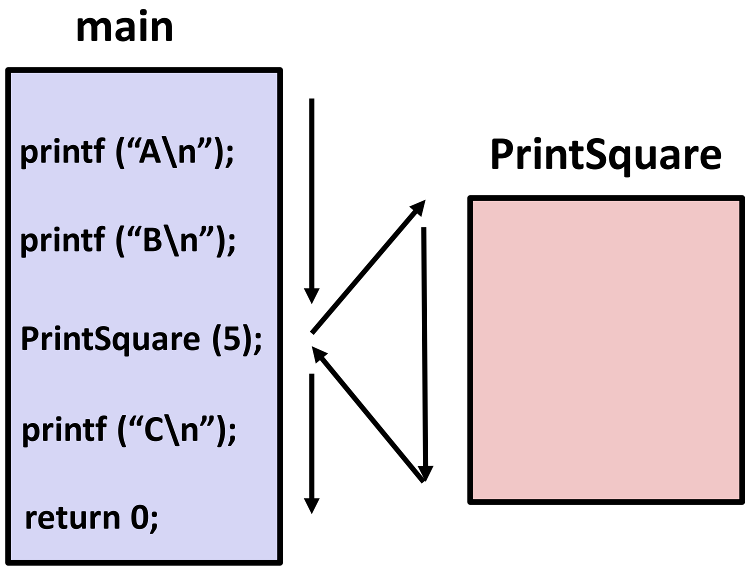
```
#include<stdio.h>

int Squares(int x)
{
    int sq_x;
    sq_x = x * x;
    return sq_x;
}

int main ()
{
    Squares(3);
    return 0;
}
```

```
#include<stdio.h>

int main ()
{
    //error or warning
    Squares(3);
    return 0;
}

int Squares(int x)
{
    int sq_x;
    sq_x = x * x;
    return sq_x;
}
```

```
#include<stdio.h>

int Square (int x);

int main ()
{
    Squares(3);
    return 0;
}

int Squares(int x)
{
    int sq_x;
    sq_x = x * x;
    return sq_x;
}
```

Sequence of Execution

```
#include <stdio.h>
void swap(int a, int b) {
    int c;
    c = b;
    b = a;
    a = c;
    printf (“a : %d, b: %d\n”, a, b);
}

void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d\n”, a, b);
     …
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d\n”, a, b);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d\n”, a, b);
}
```



main

printf (“A\n”);

printf (“B\n”);

PrintSquare (5);

printf (“C\n”);

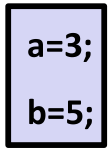
return 0;

PrintSquare

Local Variables & Parameters

```
#include <stdio.h>
void swap(int a, int b) {
    int c;
    c = b;
    b = a;
    a = c;
    printf (“a : %d, b: %d\n”, a, b);
}

void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d\n”, a, b);
     …
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d\n”, a, b);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d\n”, a, b);
}
```

Memory Space
(Stack)

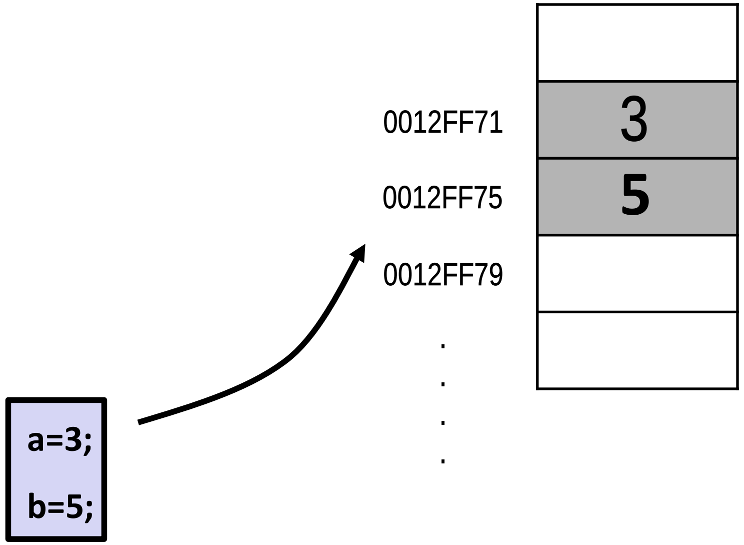


a=3;

b=5;

Local Variables & Parameters

```
#include <stdio.h>
void swap(int a, int b) {
    int c;
    c = b;
    b = a;
    a = c;
    printf (“a : %d, b: %d\n”, a, b);
}

void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d\n”, a, b);
     …
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d\n”, a, b);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d\n”, a, b);
}
```

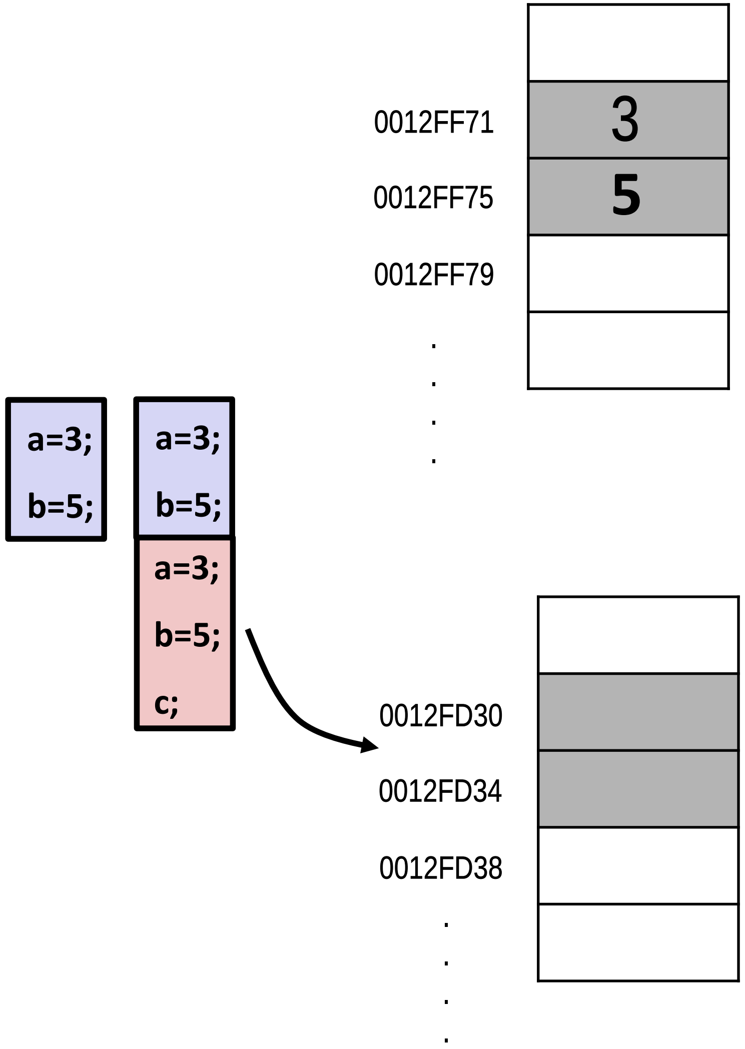


a=3;
b=5;

0012FF71 3
0012FF75 5
0012FF79

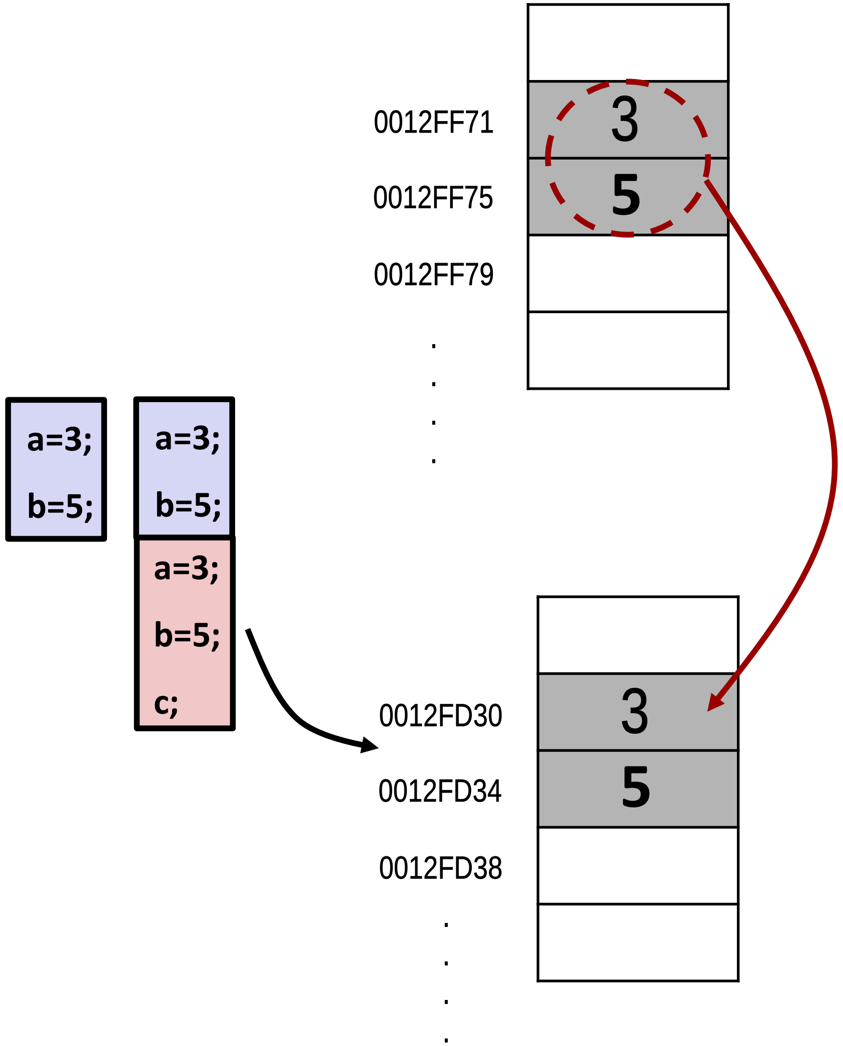
Local Variables & Parameters

```
#include <stdio.h>
void swap(int a, int b) {
    int c;
    c = b;
    b = a;
    a = c;
    printf (“a : %d, b: %d\n”, a, b);
}

void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d\n”, a, b);
     …
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d\n”, a, b);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d\n”, a, b);
}
```



Local Variables & Parameters

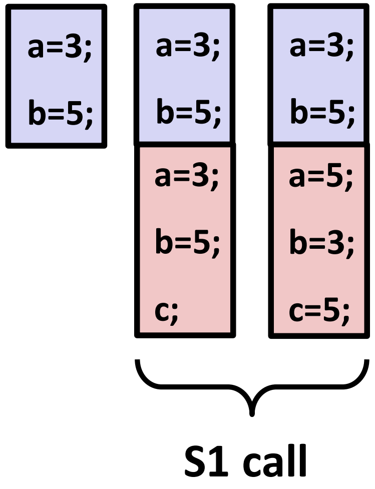
```
#include <stdio.h>
void swap(int a, int b) {
    int c;
    c = b;
    b = a;
    a = c;
    printf (“a : %d, b: %d\n”, a, b);
}

void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d\n”, a, b);
     …
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d\n”, a, b);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d\n”, a, b);
}
```



Local Variables & Parameters

```
#include <stdio.h>
void swap(int a, int b) {
    int c;
    c = b;
    b = a;
    a = c;
    printf (“a : %d, b: %d\n”, a, b);
}

void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d\n”, a, b);
     …
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d\n”, a, b);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d\n”, a, b);
}
```

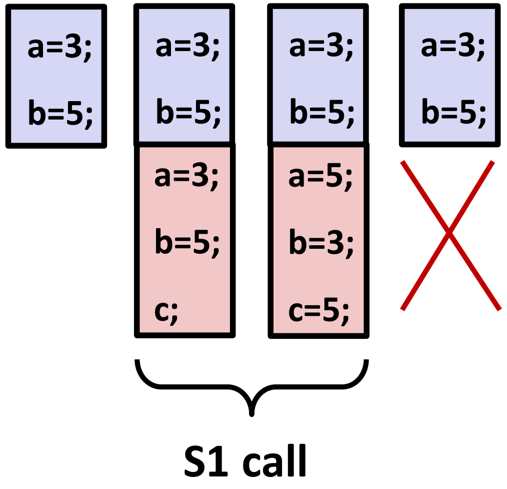
Memory Space (Stack)



Local Variables & Parameters

```
#include <stdio.h>
void swap(int a, int b) {
    int c;
    c = b;
    b = a;
    a = c;
    printf (“a : %d, b: %d\n”, a, b);
}

void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d\n”, a, b);
     …
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d\n”, a, b);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d\n”, a, b);
}
```

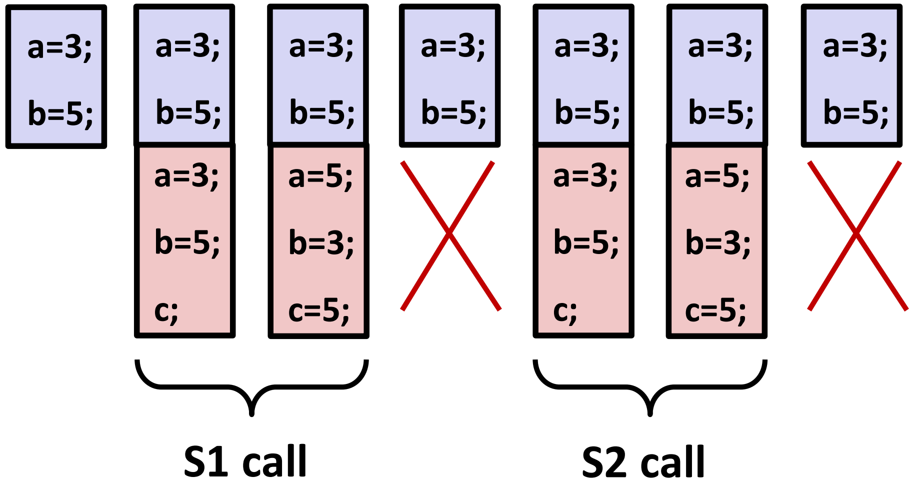
Memory Space (Stack)



Local Variables & Parameters

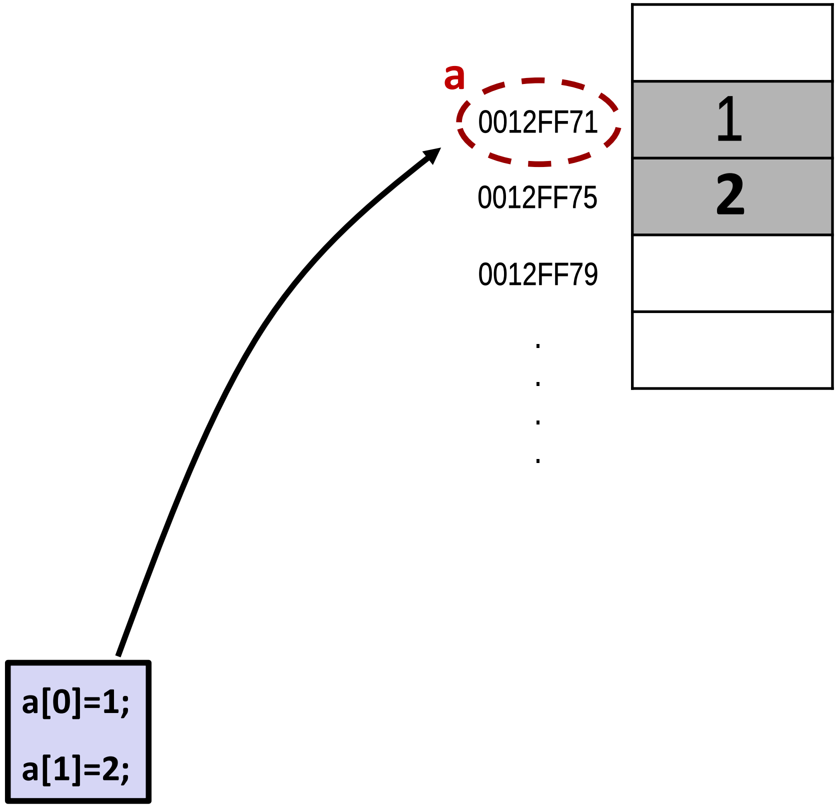
```
#include <stdio.h>
void swap(int a, int b) {
    int c;
    c = b;
    b = a;
    a = c;
    printf (“a : %d, b: %d\n”, a, b);
}

void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d\n”, a, b);
     …
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d\n”, a, b);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d\n”, a, b);
}
```

Memory Space (Stack)



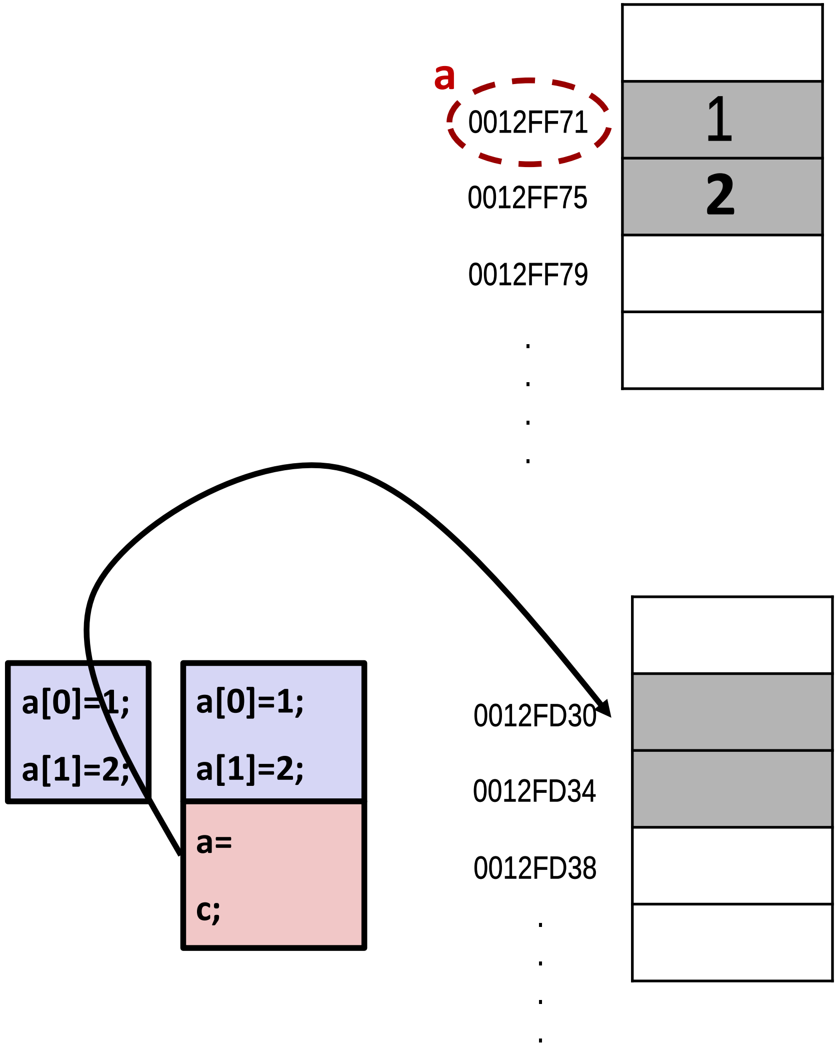
Local Variables & Parameters

```
#include <stdio.h>
void swap(int[] a) {
    int c;
    c = a[1];
    a[1] = a[0];
    a[0] = c;
    printf ("%d, %d\n", a[0], a[1]);
}

void main (void) {
     int a[2] = {1, 2};
     printf ("%d, %d\n", a[0], a[1]);
     ....
     swap(a);   //S1 call
     printf ("%d, %d\n", a[0], a[1]);
     swap(a);   //S2 call
     printf ("%d, %d\n", a[0], a[1]);
}
```



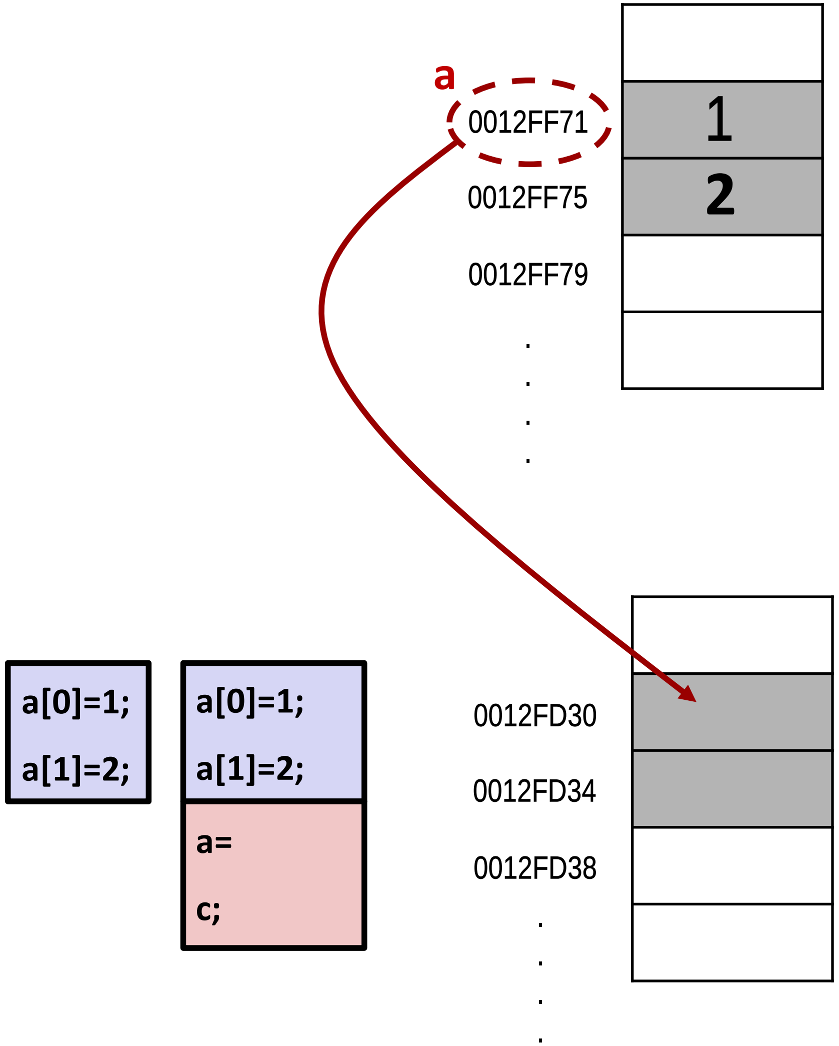
Local Variables & Parameters

```
#include <stdio.h>
void swap(int[] a) {
    int c;
    c = a[1];
    a[1] = a[0];
    a[0] = c;
    printf (“%d, %d\n”, a[0], a[1]);
}

void main (void) {
    int a[2] = {1, 2};
    printf (“%d, %d\n”, a[0], a[1]);
    ....
    swap(a);   //S1 call
    printf (“%d, %d\n”, a[0], a[1]);
    swap(a);   //S2 call
    printf (“%d, %d\n”, a[0], a[1]);
}
```



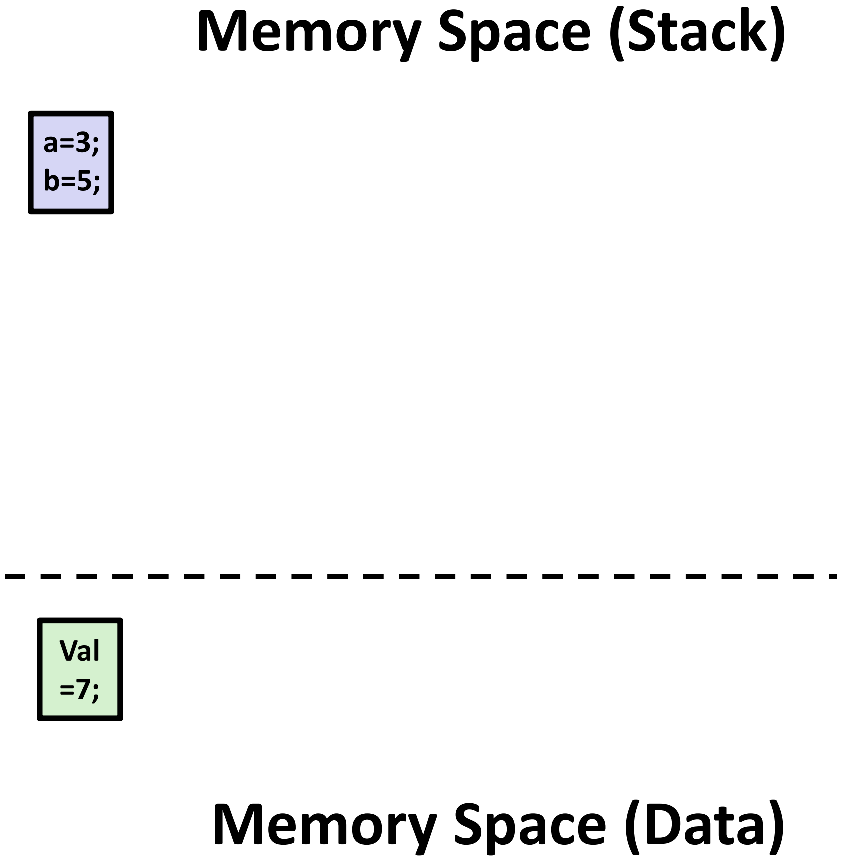
Local Variables & Parameters

```
#include <stdio.h>
void swap(int[] a) {
    int c;
    c = a[1];
    a[1] = a[0];
    a[0] = c;
    printf (“%d, %d\n”, a[0], a[1]);
}

void main (void) {
     int a[2] = {1, 2};
     printf (“%d, %d\n”, a[0], a[1]);
     ….
     swap(a);   //S1 call
     printf (“%d, %d\n”, a[0], a[1]);
     swap(a);   //S2 call
     printf (“%d, %d\n”, a[0], a[1]);
}
```



Global Variables

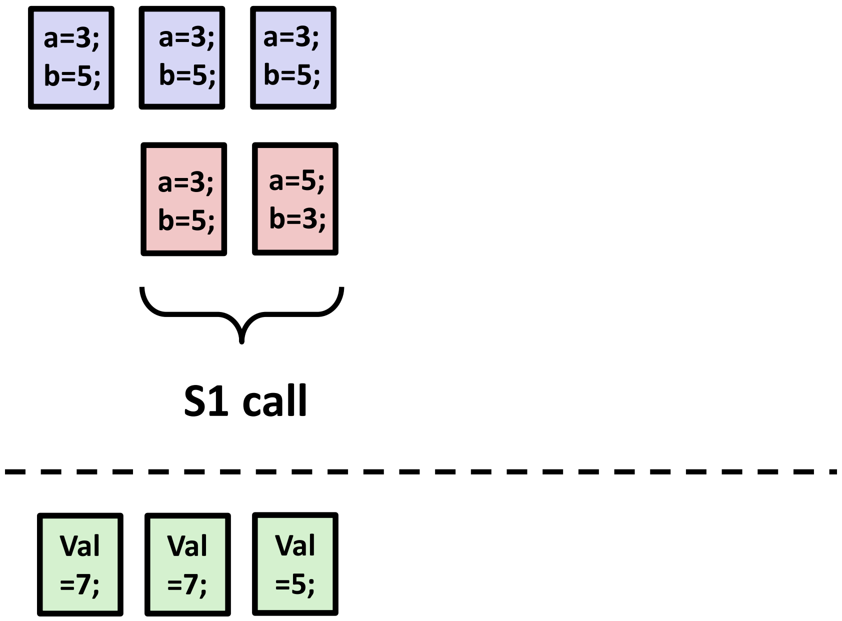
```
#include <stdio.h>

int val = 7;

void swap(int a, int b) {
    val = b;
    b = a;
    a = val;
    printf (“a : %d, b: %d\n”, a, b);
}
void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
}
```



Global Variables

```
#include <stdio.h>

int val = 7;

void swap(int a, int b) {
    val = b;
    b = a;
    a = val;
    printf (“a : %d, b: %d\n”, a, b);
}
void main (void) {
    int a=3, b=5;
    printf (“a : %d, b: %d, val: %d\n”, a, b, val);
    swap(a, b);   //S1 call
    printf (“a : %d, b: %d, val: %d\n”, a, b, val);
    swap(a, b);   //S2 call
    printf (“a : %d, b: %d, val: %d\n”, a, b, val);
}
```

Memory Space (Stack)

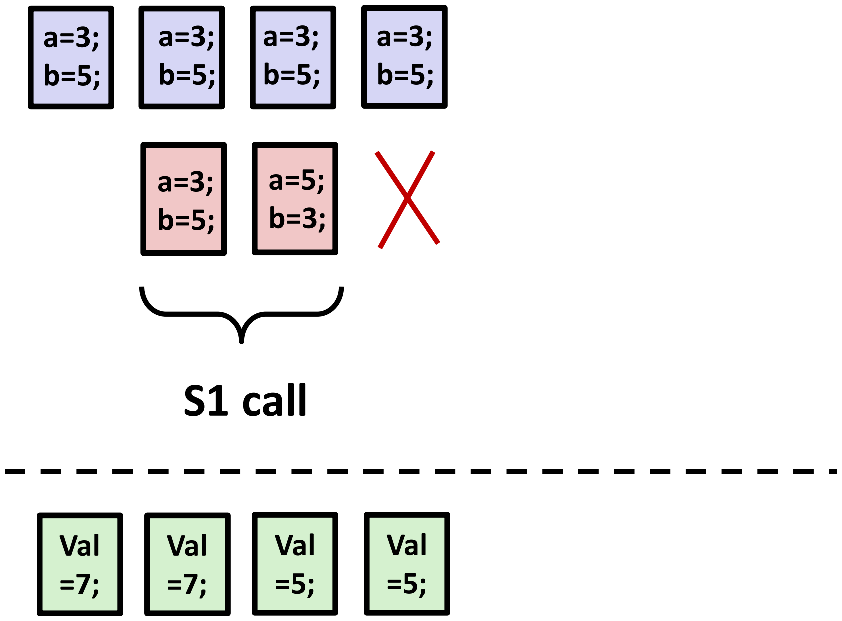


Memory Space (Data)

Global Variables

```
#include <stdio.h>

int val = 7;

void swap(int a, int b) {
    val = b;
    b = a;
    a = val;
    printf (“a : %d, b: %d\n”, a, b);
}
void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
}
```

Memory Space (Stack)



a=3; b=5;

a=3; b=5;

a=3; b=5;

a=3; b=5;

a=3; b=5;

a=5; b=3;

S1 call

Val =7;

Val =7;

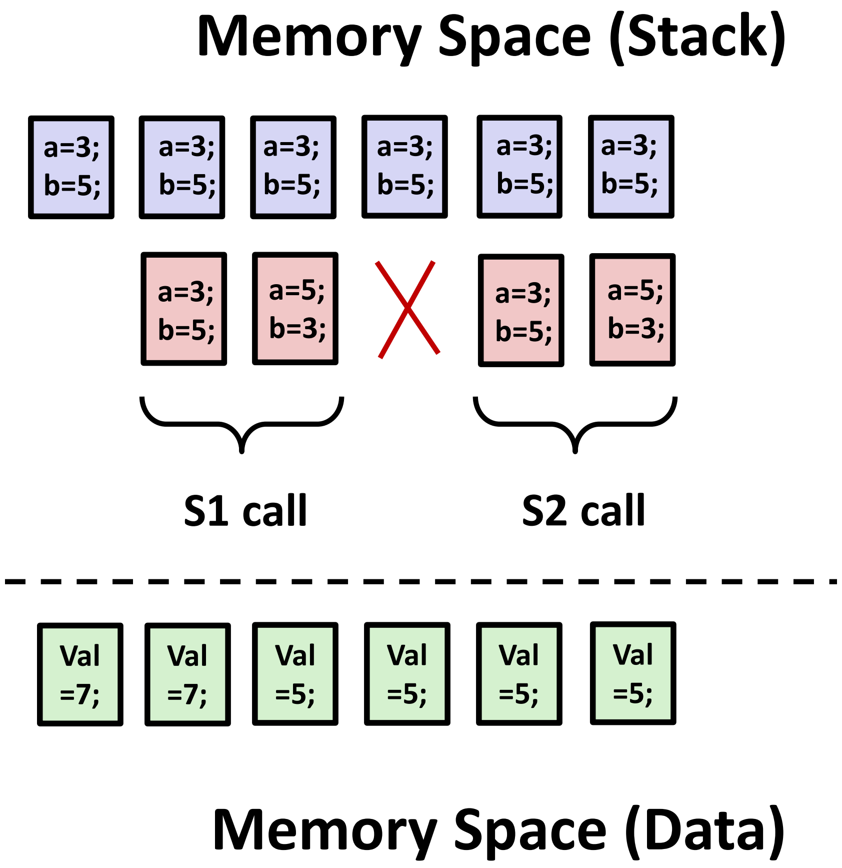
Val =5;

Val =5;

Memory Space (Data)

Global Variables

```
#include <stdio.h>

int val = 7;

void swap(int a, int b) {
    val = b;
    b = a;
    a = val;
    printf (“a : %d, b: %d\n”, a, b);
}
void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
}
```



Memory Space (Stack)

a=3; b=5;	a=3; b=5;	a=3; b=5;	a=3; b=5;	a=3; b=5;	a=3; b=5;
	a=3; b=5;	a=5; b=3;	X	a=3; b=5;	a=5; b=3;
Val =7;	Val =7;	Val =5;	Val =5;	Val =5;	Val =5;

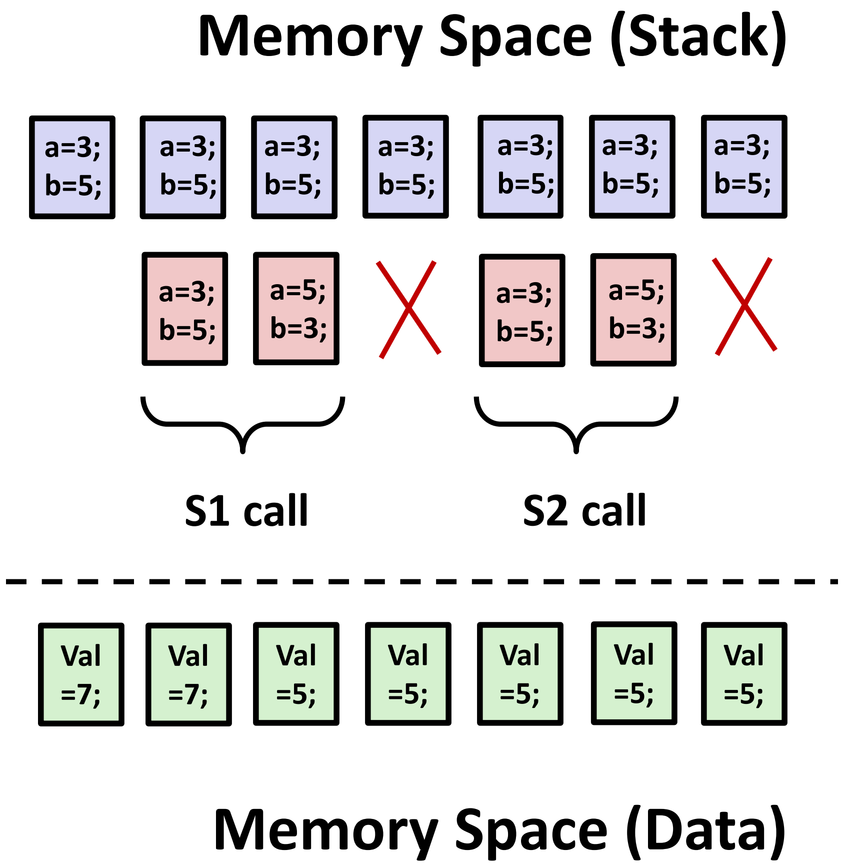
S1 call

S2 call

Memory Space (Data)

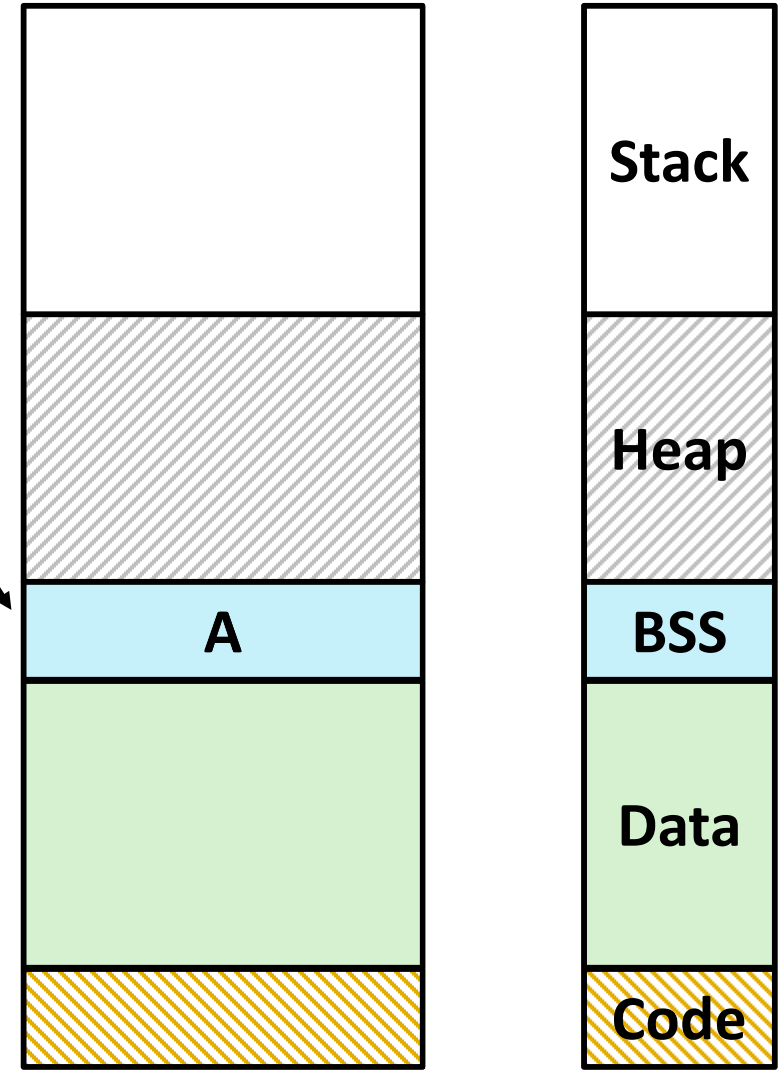
Global Variables

```
#include <stdio.h>

int val = 7;

void swap(int a, int b) {
    val = b;
    b = a;
    a = val;
    printf (“a : %d, b: %d\n”, a, b);
}
void main (void) {
     int a=3, b=5;
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
     swap(a, b);   //S1 call
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
     swap(a, b);   //S2 call
     printf (“a : %d, b: %d, val: %d\n”, a, b, val);
}
```



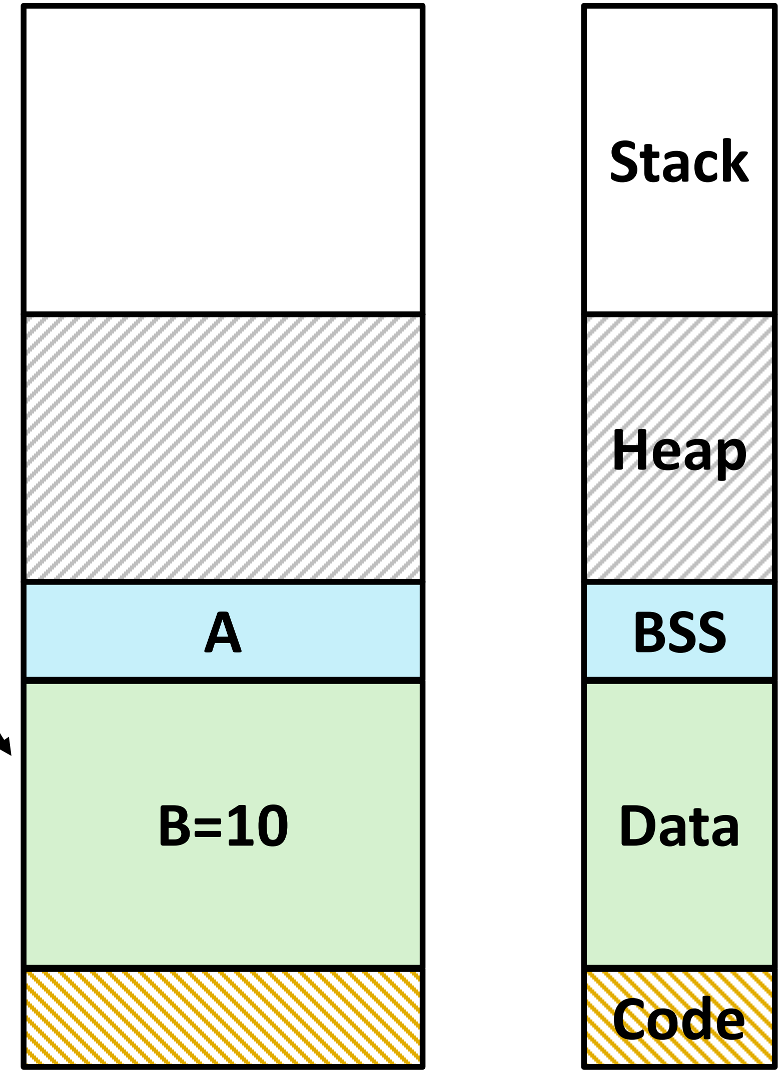
Variable Location in Memory

```
#include <stdio.h>

int A;
int B = 10;
void func(int C) {
    int D = 0;
    printf (“%d, %d\n”, C, D);
}
void main (void) {
     int E = 7;
     printf (“%d\n”, E);
     func (E);
}
```



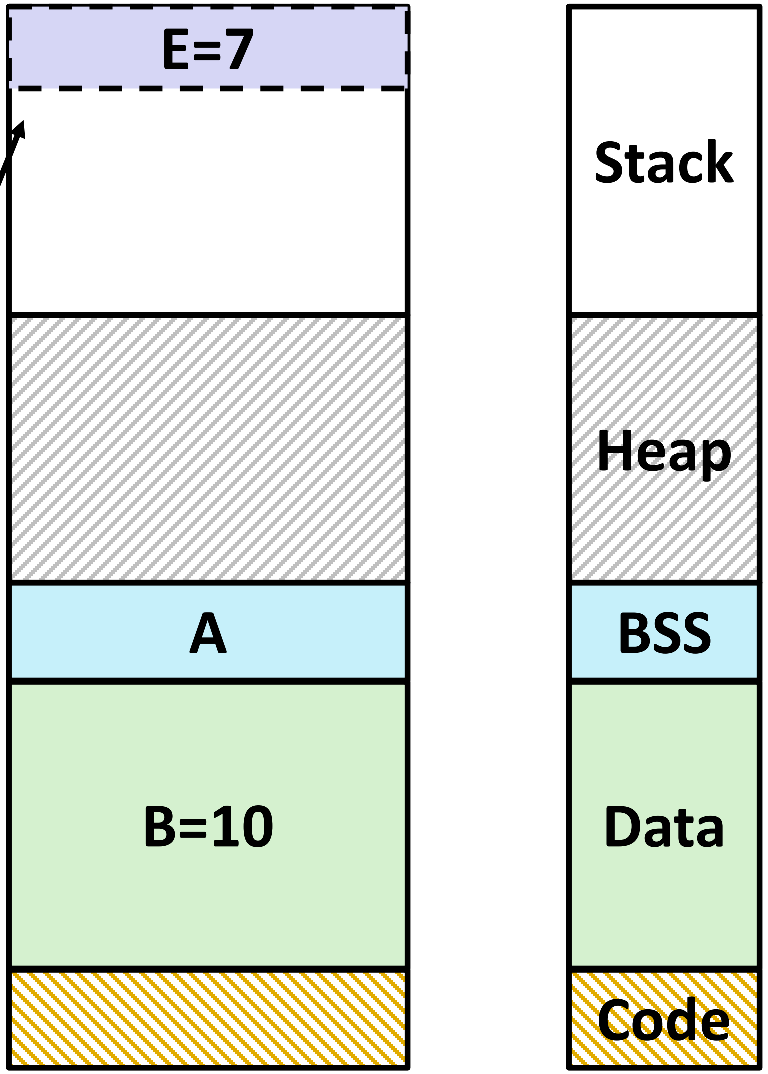
Variable Location in Memory

```
#include <stdio.h>

int A;
int B = 10;
void func(int C) {
    int D = 0;
    printf (“%d, %d\n”, C, D);
}
void main (void) {
     int E = 7;
     printf (“%d\n”, E);
     func (E);
}
```



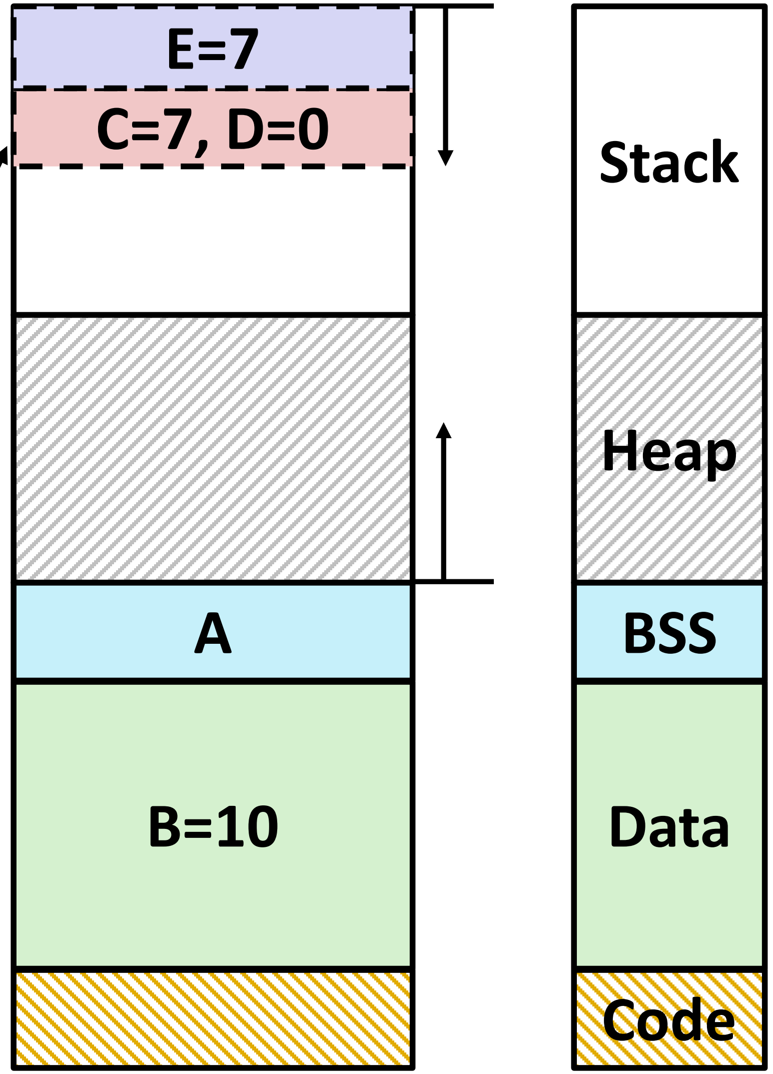
Variable Location in Memory

```
#include <stdio.h>

int A;
int B = 10;
void func(int C) {
    int D = 0;
    printf (“%d, %d\n”, C, D);
}
void main (void) {
     int E = 7;
     printf (“%d\n”, E);
     func (E);
}
```



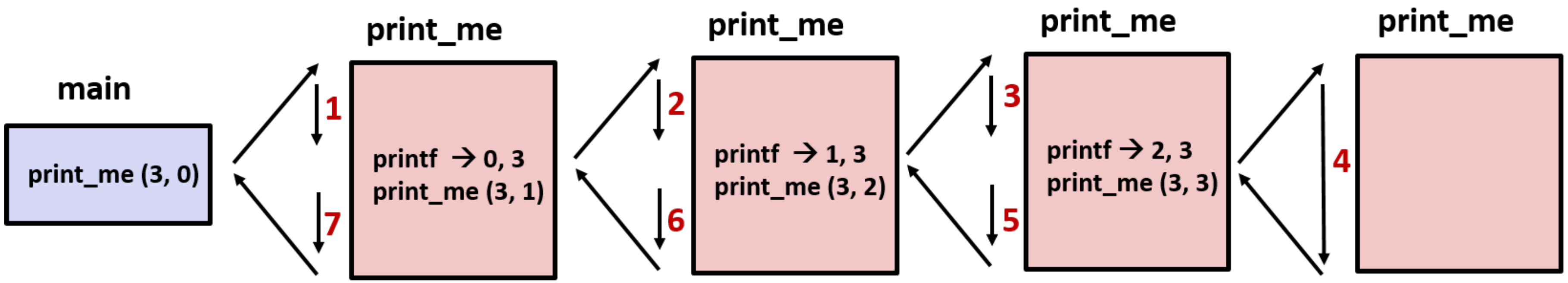
Variable Location in Memory

```
#include <stdio.h>

int A;
int B = 10;
void func(int C) {
    int D = 0;
    printf (“%d, %d\n”, C, D);
}
void main (void) {
     int E = 7;
     printf (“%d\n”, E);
     func (E);
}
```



Recursion

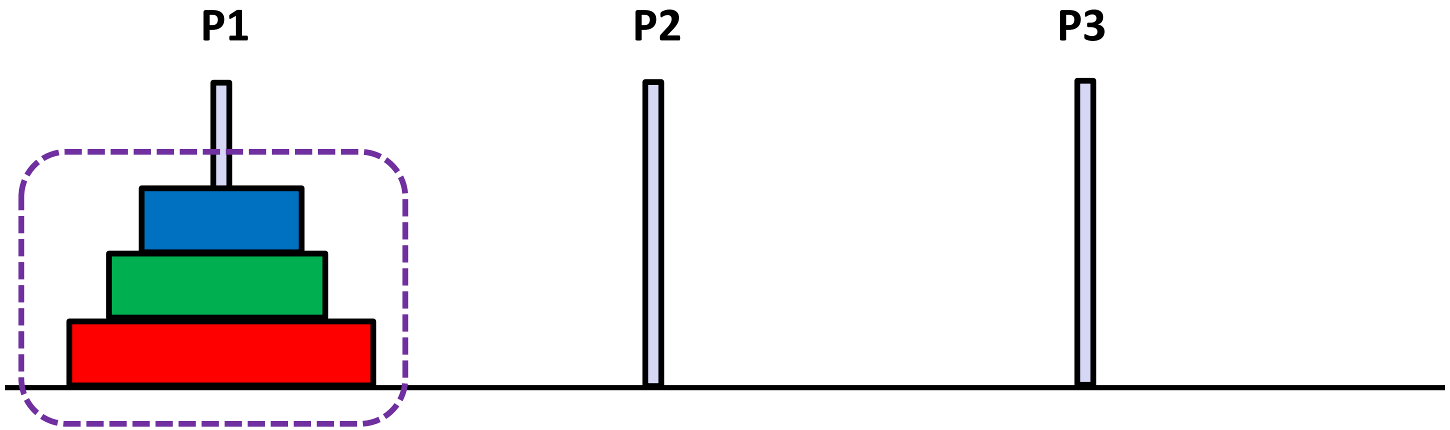
```
void print_me (int j, int depth)
{
    if(depth < j ) {
        printf (“Recursion! Depth =  %d, j = %d \n”, depth, j);
        printf_me (j, ++depth);
    }
}
```



Recursion

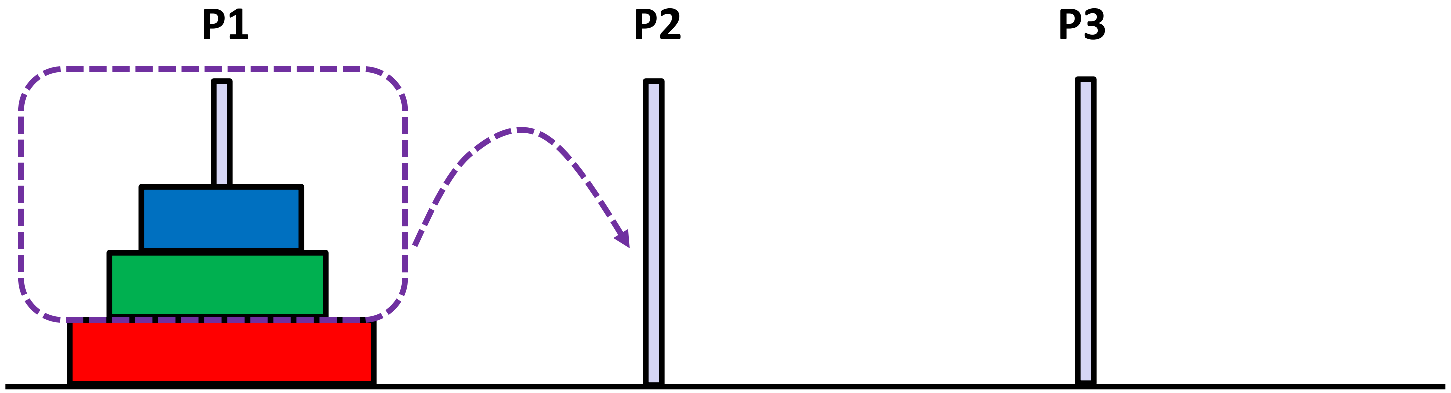
```
/* Don’t run this code */
void infinite_recursion ()
{
    printf(“Infinite loop ! \n”);
    infinite_recursion ();
}
```

Recursion Example – Hanoi Tower



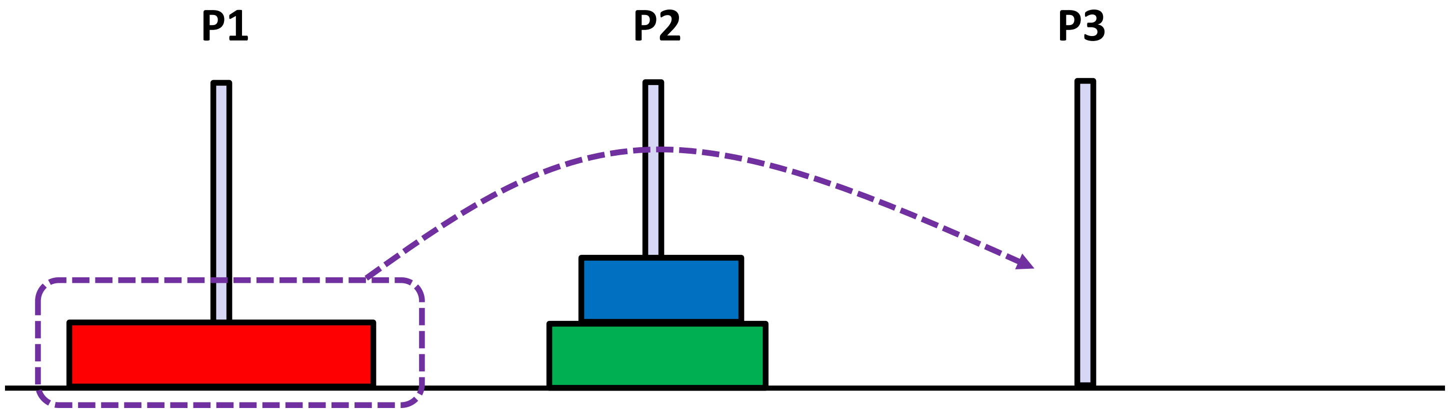
Objective : Move 3 disks from P1 to P3

Step-by-Step

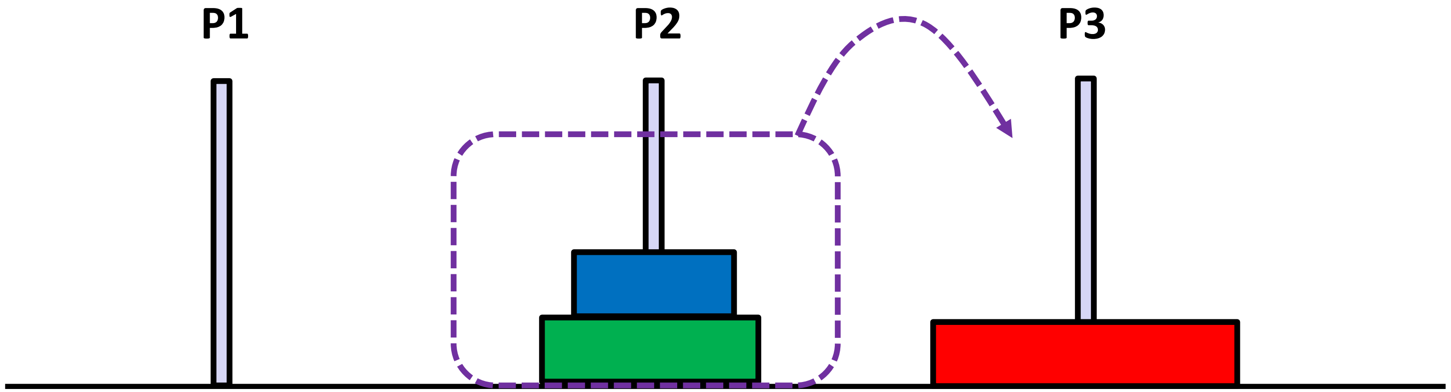


Sub objective : Move 2 disks from P1 to P2

Recursion Example – Hanoi Tower

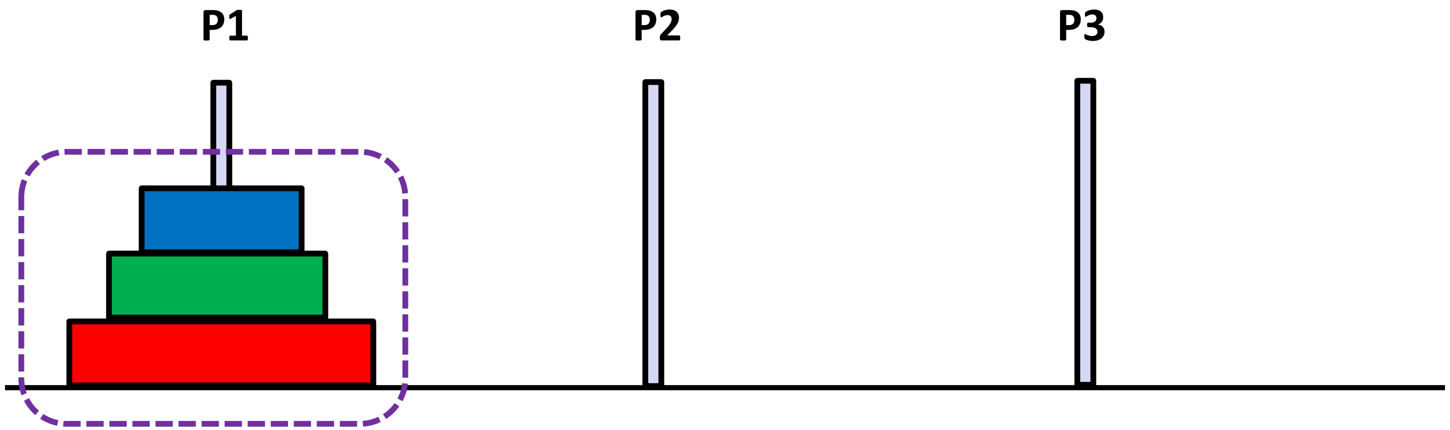


Sub objective : Move the largest from P1 to P3

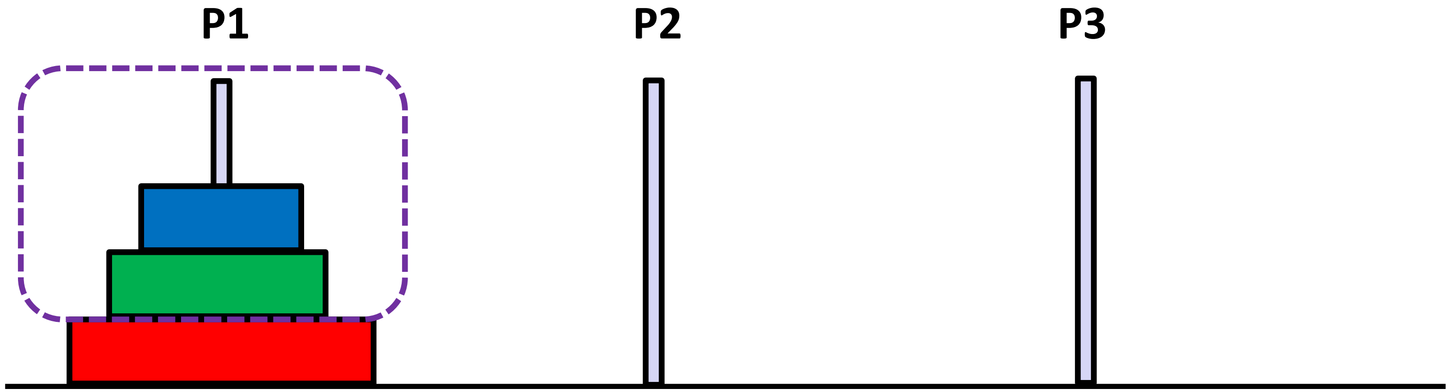


Sub objective : Move 2 disks from P2 to P3

Recursion Example – Hanoi Tower



Objective : Move n disks from P1 to P3



Objective : Move n-1 disks from P1 to P2
Move the largest from P1 to P3
Move n-1 disks from P2 to P3

Recursion Example – Hanoi Tower

```
#include <stdio.h>

void towerHanoi (int n, char from, char to, char aux)
{
    if (n== 1)
    {
        printf(“n Move disk 1 from rod %c to %c\n”, from, to);
        return;
    }
    towerHanoi (n-1, from, aux, to);
    printf (“n Move disk %d from rod %c to %c\n”, from, to);
    towerHanoi (n-1, aux, to, from);
 }

 void main()
 {
     int disks = 4;
     towerHanoi (disk, ‘A’, ‘B’, ‘C’);
 }
```

What we have covered today

- Definition/Declaration of Functions
- Sequence of Execution when functions are called
- Local/global variables
- Parameters – Value/Address Copy
- Recursion

Q and A