Passing Pointers to a Function

Mohammed Bahja
School of Computer Science
University of Birmingham

Pass-by-reference and Pass-by-value

- We have seen how to pass data objects to a function as arguments. This technique is called ‘pass-by-value’.

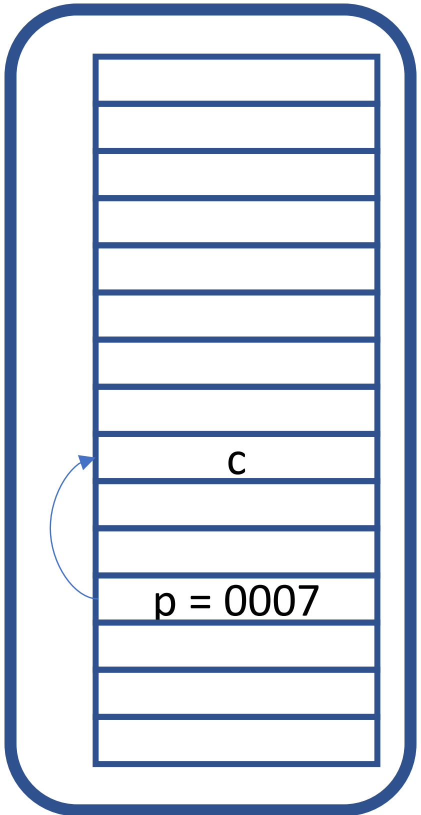
- We can pass pointers to a function as arguments.
- This is known as ‘pass-by-reference’.

```
foo(int c){
   c=c*5;
   ...
}
int main(){
   int c=5;
   foo(c);
}
```

Passing object to foo( ).

```
foo(int *p){
   *p=*p*5;
   ...
}
int main(){
   int c=5;
   int *p = &c;
   foo(p);
}
```

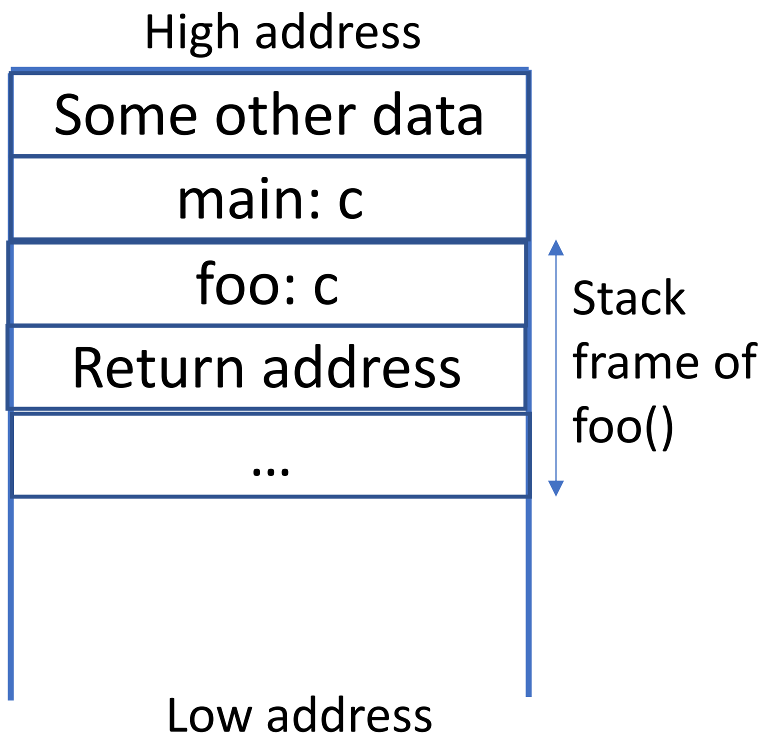
Passing pointer to foo( )



Pass-by-reference vs Pass-by-value: difference

```
foo(int c){
   c=c*5; // Scope is foo
   ...
}
int main(){
   int c=5;
   foo(c);
}
```

Example of pass-by-value



Stack during function calls

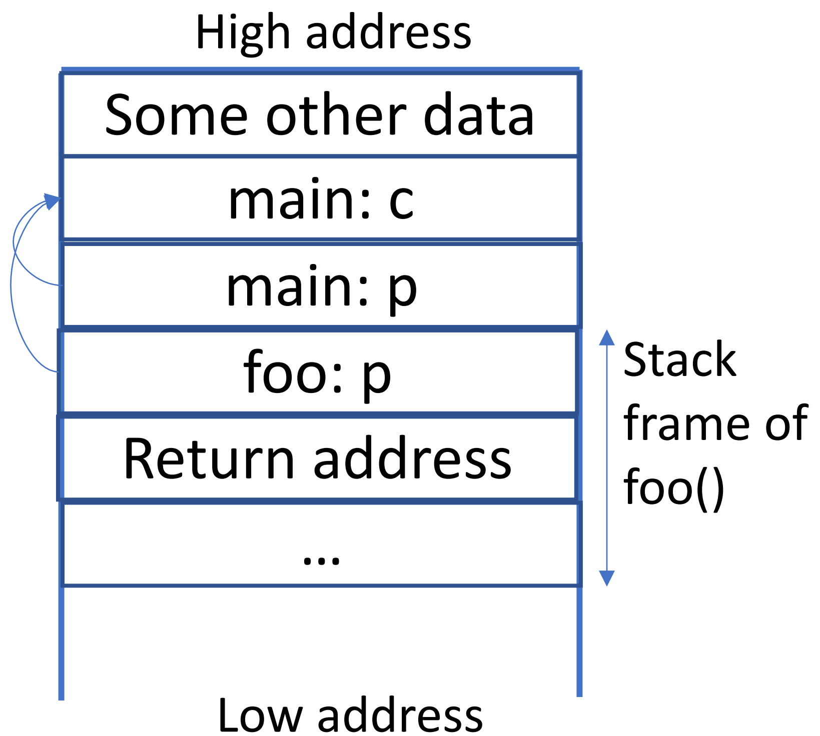
Consequences:

- foo() gets a local copy of c.
  So, c=c*5=25 happens only within foo().
- main() still sees c=5.

Pass-by-reference vs Pass-by-value: difference

```
foo(int *p){
   *p=*p*5;
   ...
}
int main(){
   int c=5;
   int *p = &c;
   foo(p);
}
```

Example of pass-by-reference



Stack during function calls

Consequences:

- foo() gets a local copy of p which contains the address of c. So, *p=*p*5=25 updates the memory location where c is stored.
- Both foo() and main() see c=25.

Example: swapping two integers

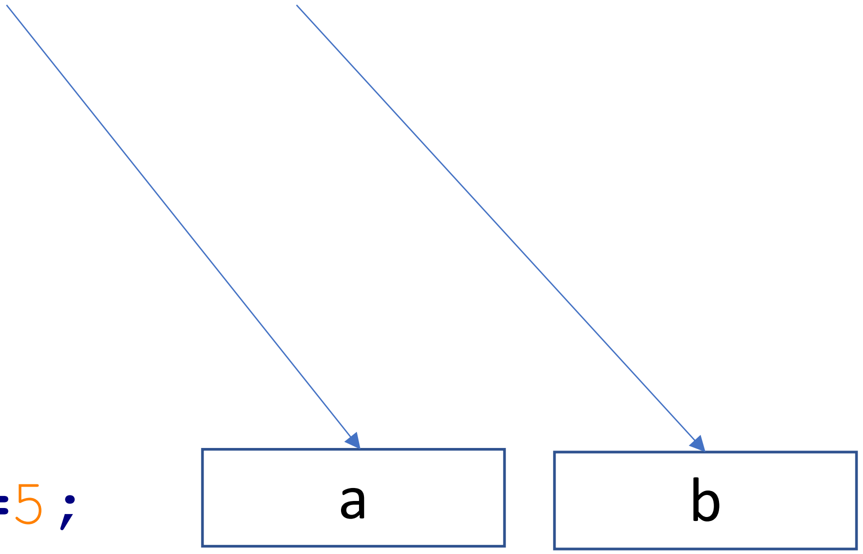
```
void swap(int x, int y){
    int temp;
    temp = x;
    x = y;
    y = temp;
}
int main(){
    int a=4, b=5;
    swap(a, b);
    printf("a=%d b=%d", a, b);
    return 0;
}
```

Changes are local,
not visible from main().

The program will print a=4 and b=5.

Example: swapping two integers

```
void swap(int *x, int *y){
     int temp;
     temp = *x;
     *x = *y;
     *y = temp;
}
int main(){
     int a=4, b=5;
     swap(&a, &b);
     printf("a=%d b=%d", a, b);
     return 0;
}
```



The program will print swapped values, i.e. a=5 and b=4.

Returning pointer from function

- A function can return a pointer.

```
int *foo(...) // Returns pointer to an int

char *foo(...) // Returns pointer to a char

float *foo(...) // Returns pointer to a float
```

Returning pointer from function

- A function can return a pointer.

  Example: Find the maximum value and return the pointer.

```
int *max(int *a, int *b){
     if(*a > *b) return a;
     else return b;
}
int main(){
     int a=4, b=5;
     int *c;
     c=max(&a, &b);
     printf("Max value=%d", *c);
     return 0;
}
```

Returning pointer from function: pitfalls

Careful: Never return pointer to a local variable.

```
int *max(int *a, int *b){
      int temp;
      if(*a > *b) temp=*a;
      else temp=*b;
      return &temp
}

int main(){
      int a=4, b=5;
      int *c;
      c=max(&a, &b);
      printf("Max value=%d", *c);
      return 0;
}
```

temp is a local object.

After function call, temp doesn’t exist.

But c points to temp.

So, c points to an object which does not exist.