Memory Management

- there are **2** main types of memory: **stack memory** and **heap memory**
- the **stack memory** is a special region in the **RAM** (*random access memory*)
- this is a special data type (stack) that store the active functions and local variables as well
- this is how Python knows where to return after finish execution of a given function

Memory Management

- there are **2** main types of memory: **stack memory** and **heap memory**
- the **heap memory** is a special region in the **RAM** (*random access memory*) as well
- the size of the heap memory is way larger than that of the stack memory (we can store more items)
- **objects** are stored on the heap memory

Memory Management

STACK MEMORY	HEAP MEMORY
small size	large size
fast access	slow access
stores function calls and local variables	stores objects
no fragmentation	may become fragmented

Memory Management

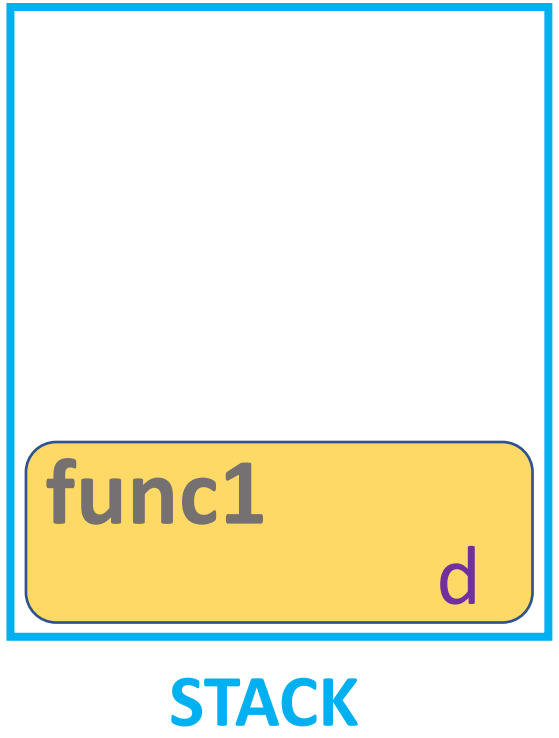
```
public void func1() {

}
```

in the **Stack** a frame will be created
from the main method

Memory Management

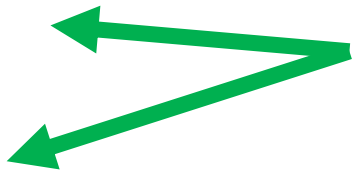
```
public void func1() {
        int d = 10;
}
```

a local variable **d** in **func1()** will also be created in the main method's frame in the stack memory.

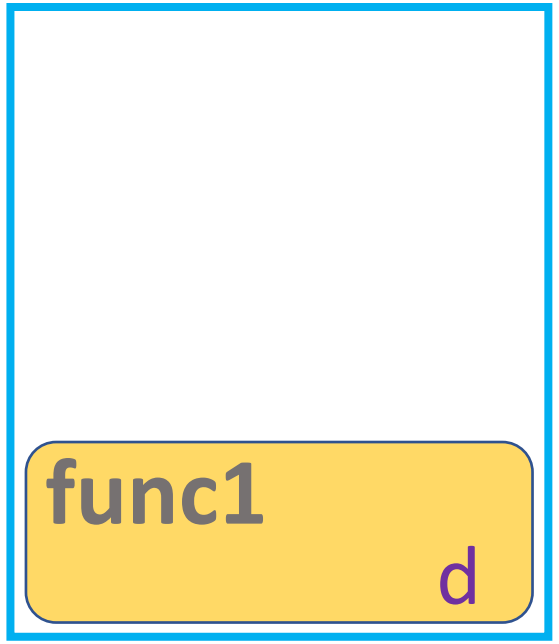


Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {

}
```



the **func1()** is calling **func2()** function

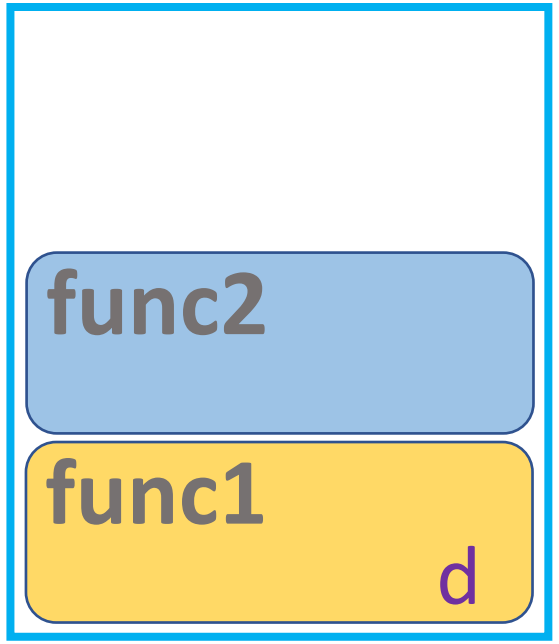


STACK

Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {

}
```

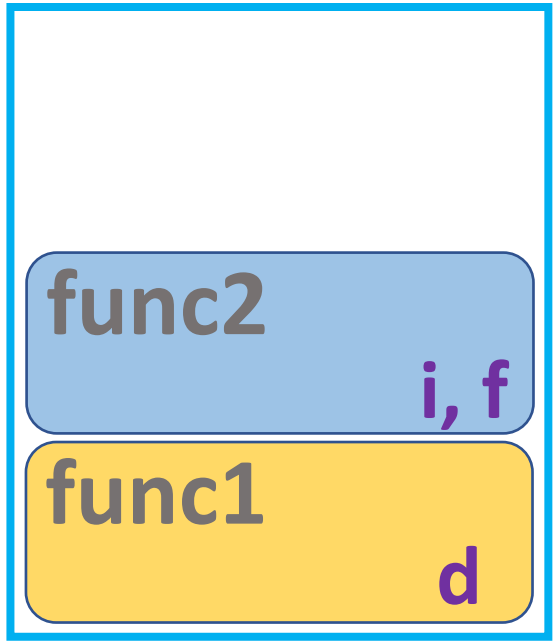
in the **stack** a new frame will be created for **func2()** on the top of the **func1()** function's frame



Memory Management

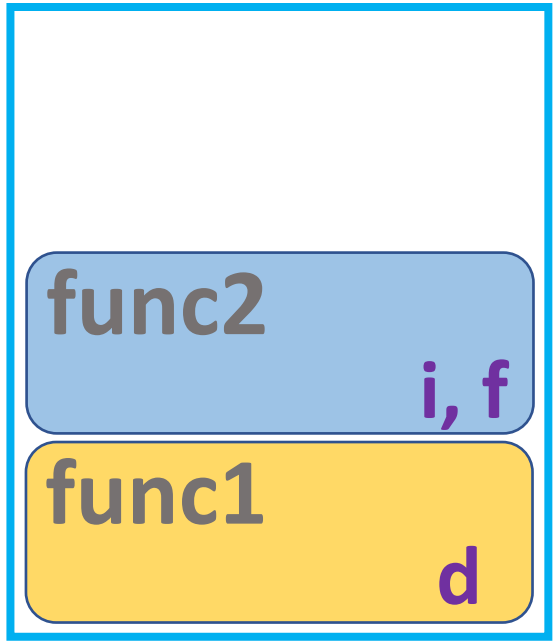
```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
}
```

in the **stack** a new frame will be created for **func2()** on the top of the **func1()** function's frame



Memory Management

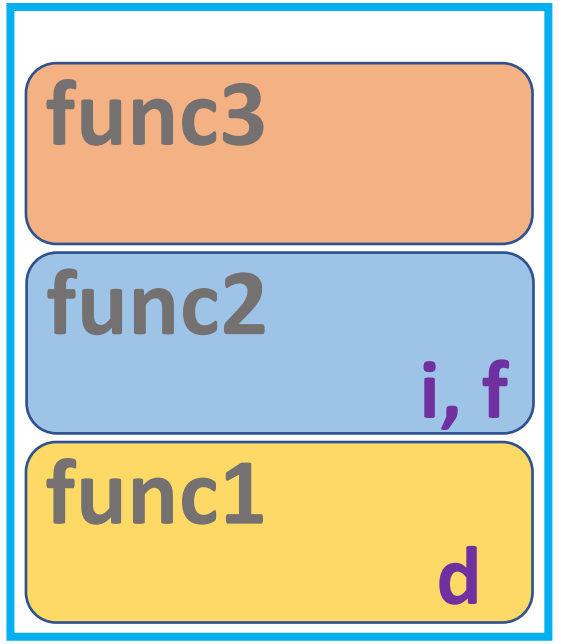
```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {

}
```



STACK

Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {

}
```

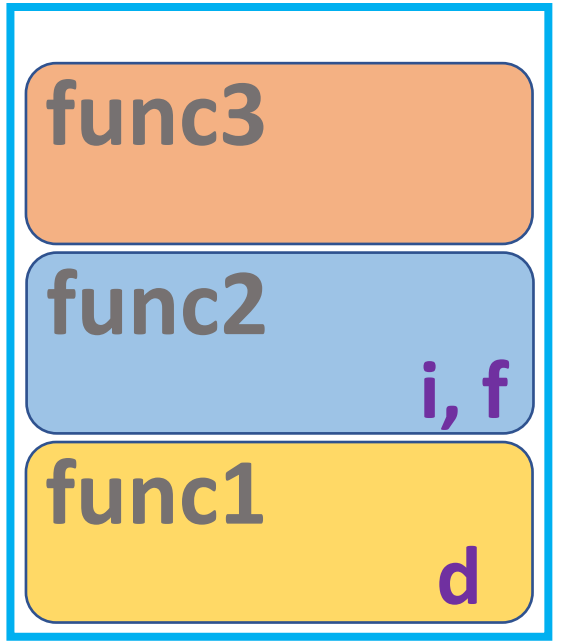


on the top of the stack a frame for **func3()** is created

Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

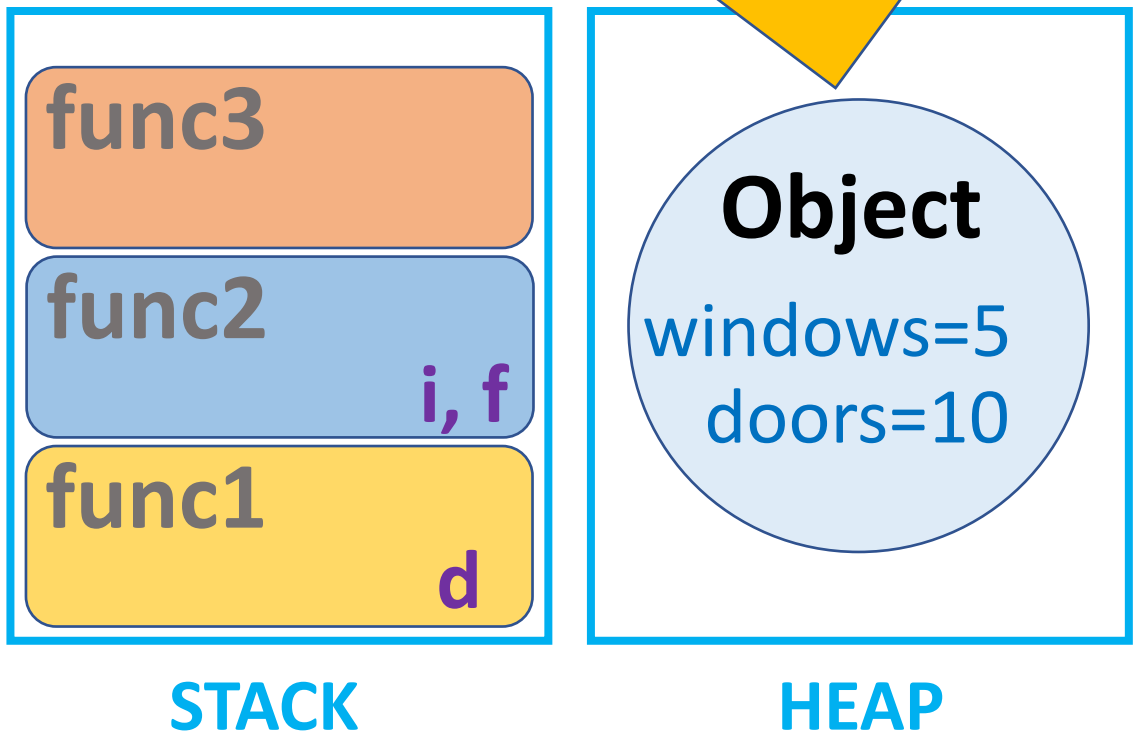


STACK

Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i)
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
        int windows = 5;
        int doors = 10;
}
```

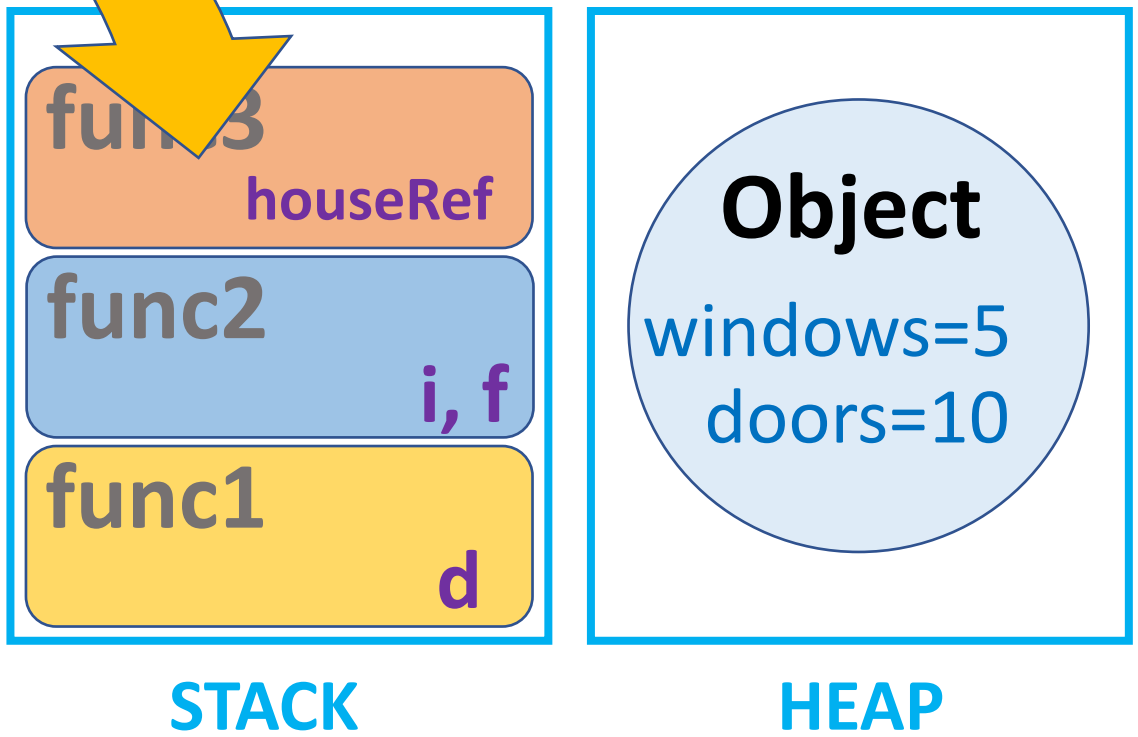


Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void f     (int i) {
        float f     f;
        func
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

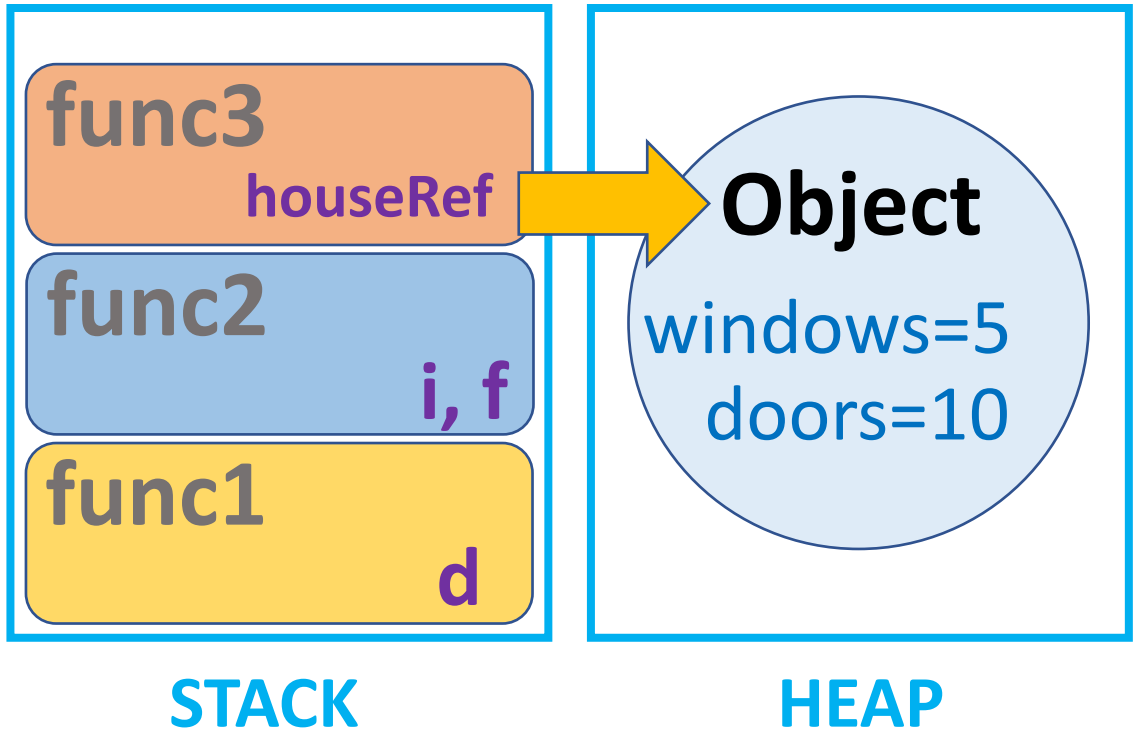
the reference variable called **houseRef** is created in the stack memory inside func3's frame



Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

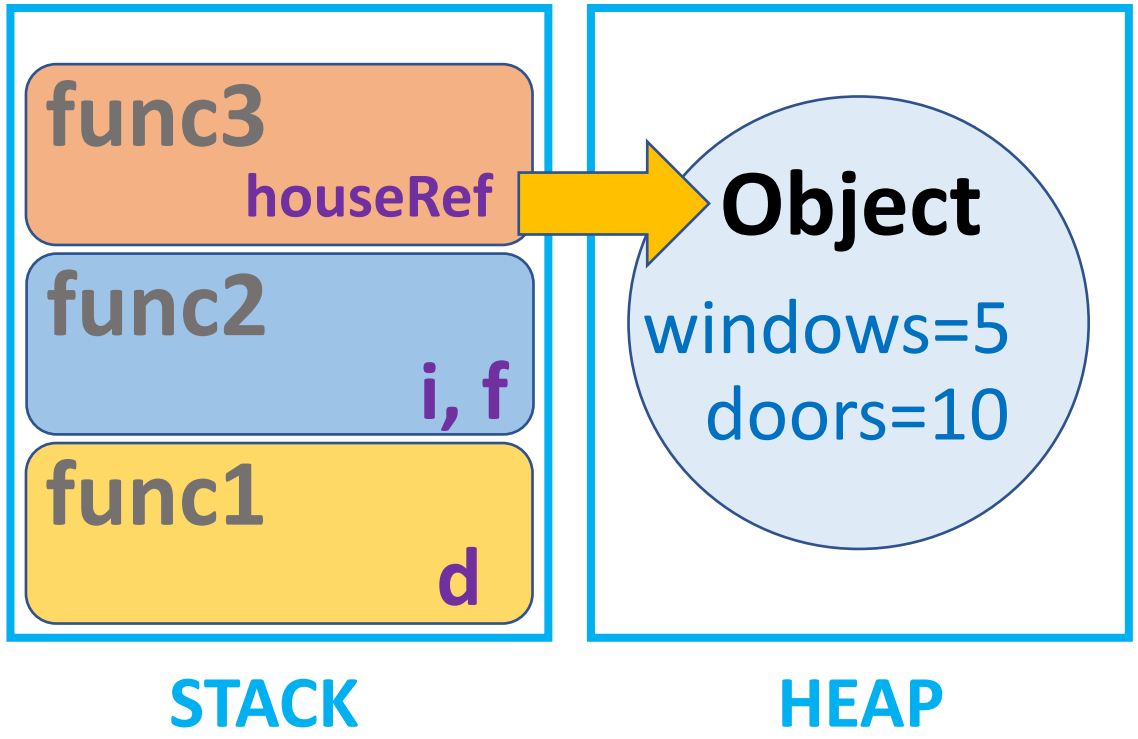


Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

when **func3()** execution is completed, the flow of the control will go back to **func2()**

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

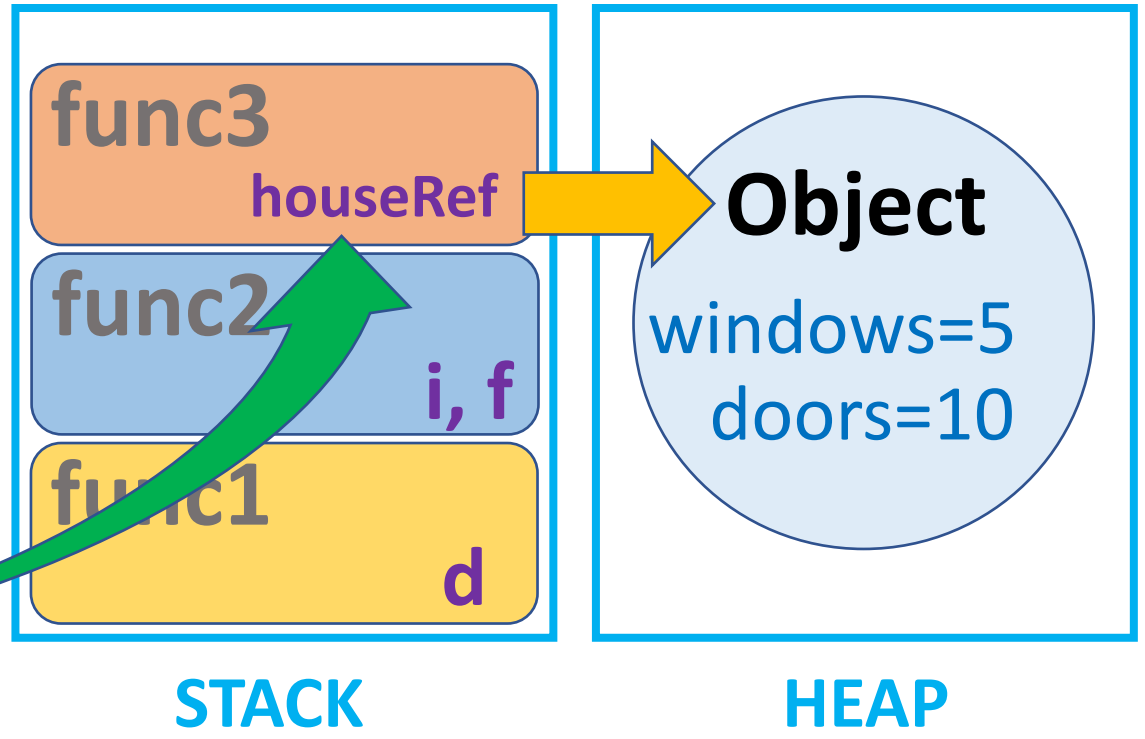


Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

because **func3()** is completed,
it is flushed out of the stack

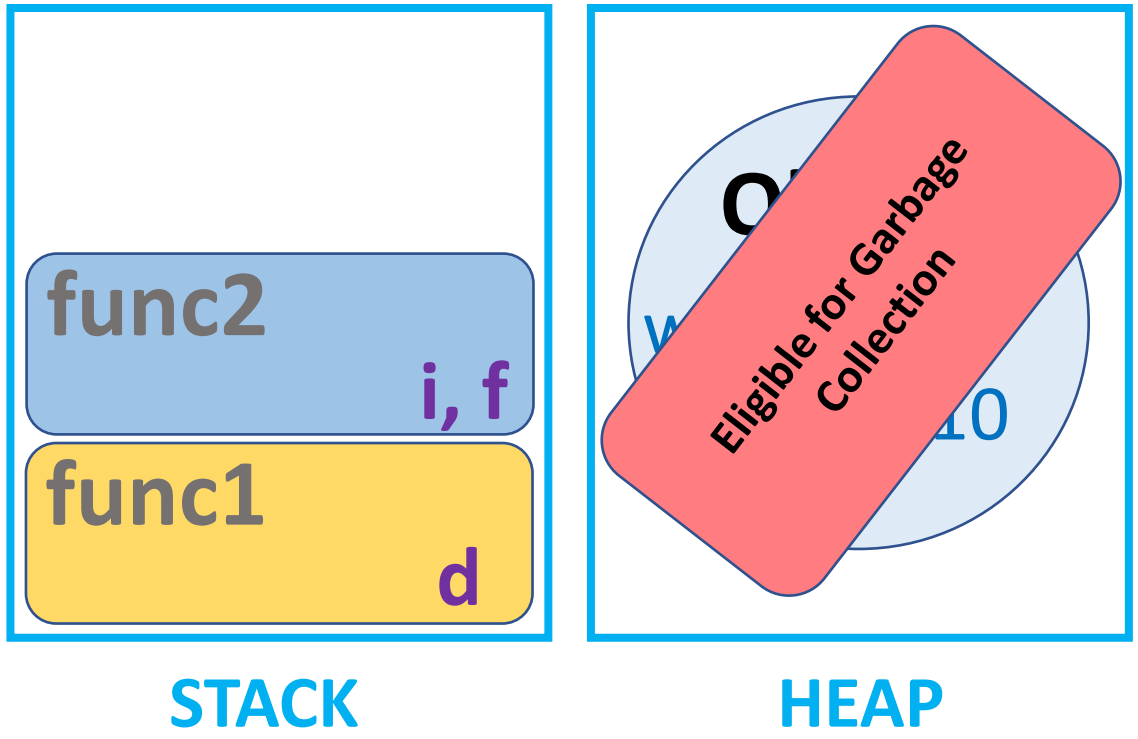


Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

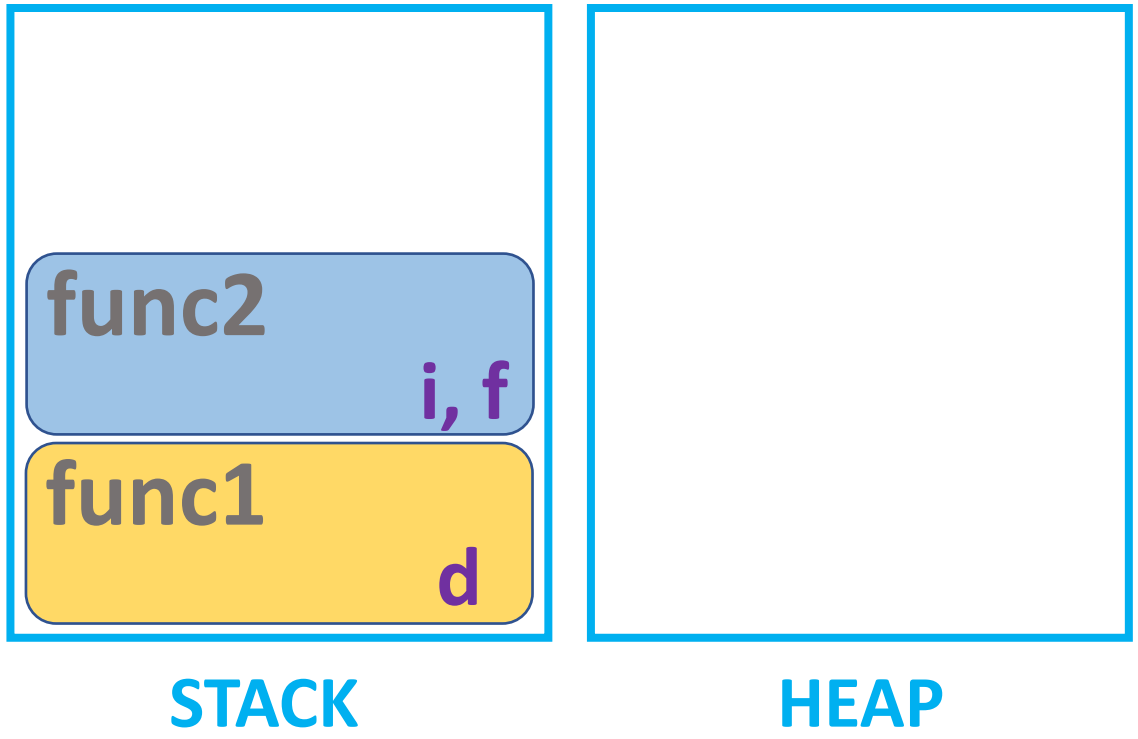
since the **houseRef** reference variable is no longer pointing to the **Object** it can be removed



Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

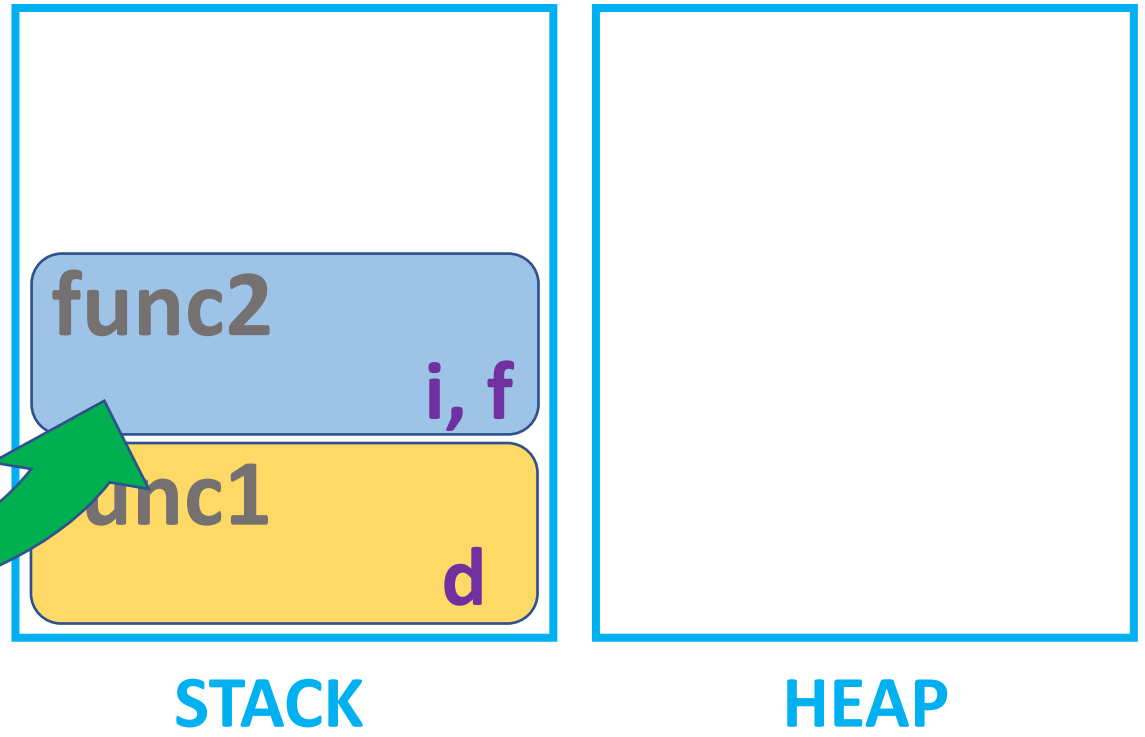


Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

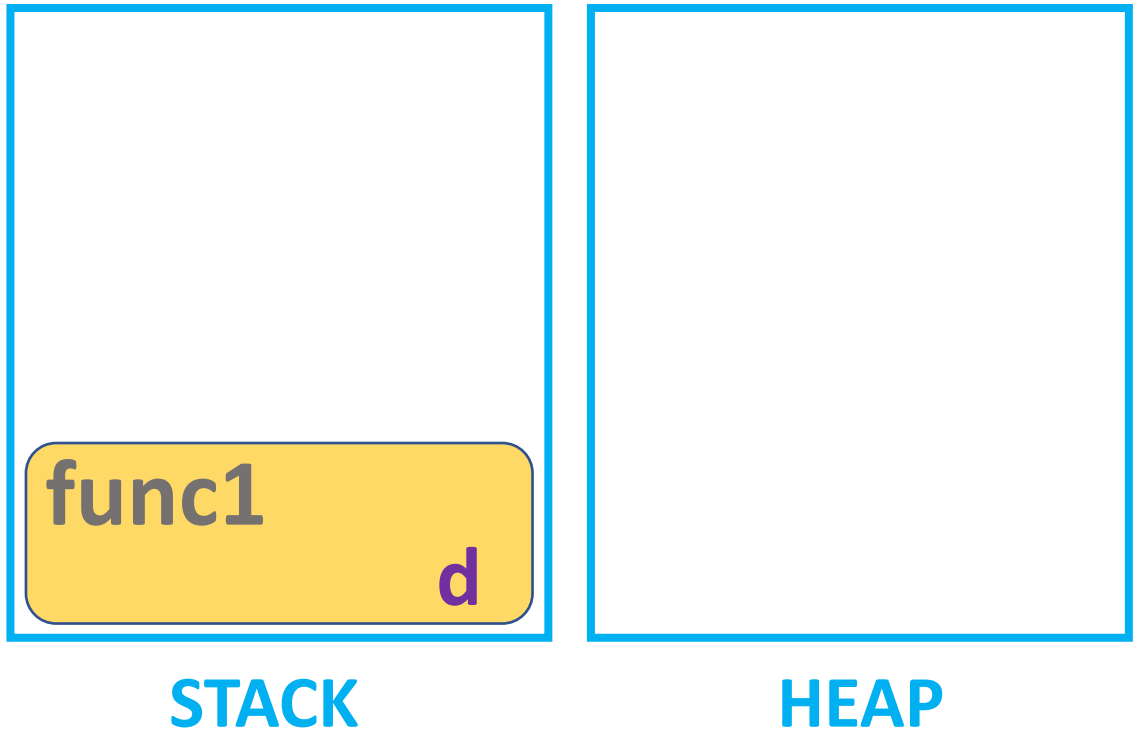
because **func2()** is completed, it is flushed out of the stack



Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```

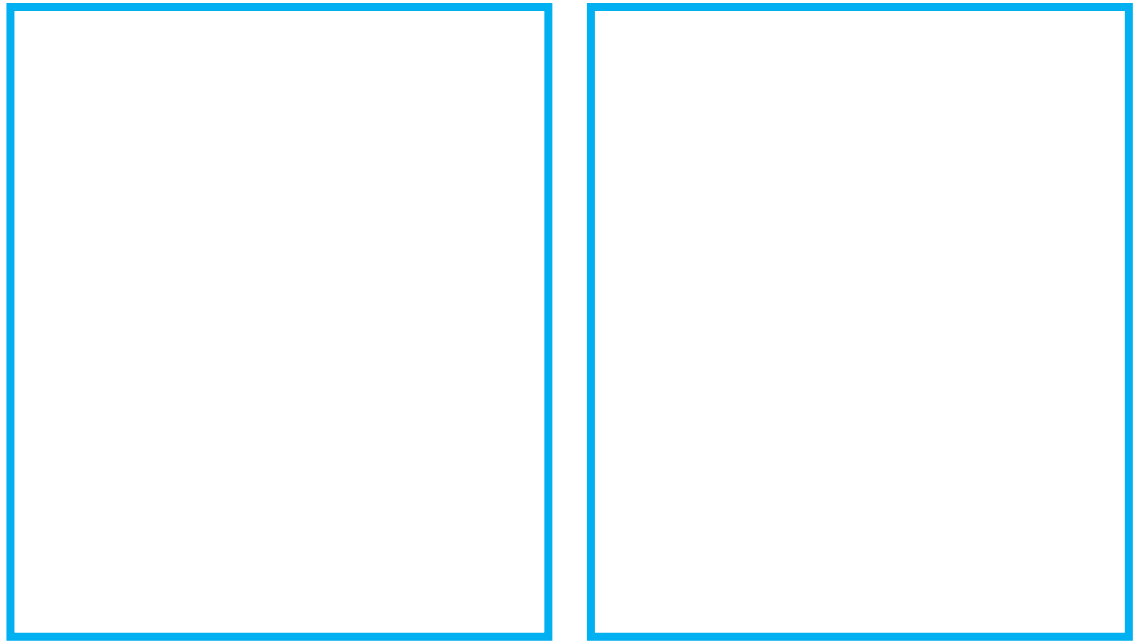


Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

because **func1()** is completed, it is flushed out of the stack

```
public class House {
    int windows = 5;
    int doors = 10;
}
```



Memory Management

```
public void func1() {
        int d = 10;
        func2(20);
}

public void func2(int i) {
        float f = 30f;
        func3();
}

public void func3() {
        houseRef = new House();
}
```

```
public class House {
    int windows = 5;
    int doors = 10;
}
```