CS Summer Challenge

Day 0x2 / 0x3

How does a code work?

How does code work?

Variable Scopes

Variable **Scope**

- When a source code file is translated into binary code, **both instructions and data are translated.** Each are allocated their own memory slots in the final executable binary file.
  - Instructions: *function calls, function definitions, and assignment operations with the form ‘ x = y +/-* z ’.*
  - Data: *all declared variables.*
- **An instruction will be able to use data if the data’s location in the binary is known to it.**
- Data can have 1) a **global** location or 2) a **local** location.
  - *Global data can be located by all instructions of the binary file.*
  - *Local data can be located by the instructions in the block of code that created it.*

```
x1 = 0
x2 = 0
result = 0

def get_input():
    x1 = int(input("Enter x1: "))
    x2 = int(input("Enter x2: "))

def add(x, y):
    sum = x + y
    return sum

def subtract(x, y):
    diff = x - y
    return diff
...
get_input()
result = add(x1, x2)
```

add() Code Block

subtract() Code Block

Global Code Block

```
x1 = 0
x2 = 0
result = 0

def get_input():
    x1 = int(input("Enter x1: "))
    x2 = int(input("Enter x2: "))

def add(x, y):
    sum = x + y
    return sum

def subtract(x, y):
    diff = x - y
    return diff
...
get_input()
result = add(x1, x2)
```

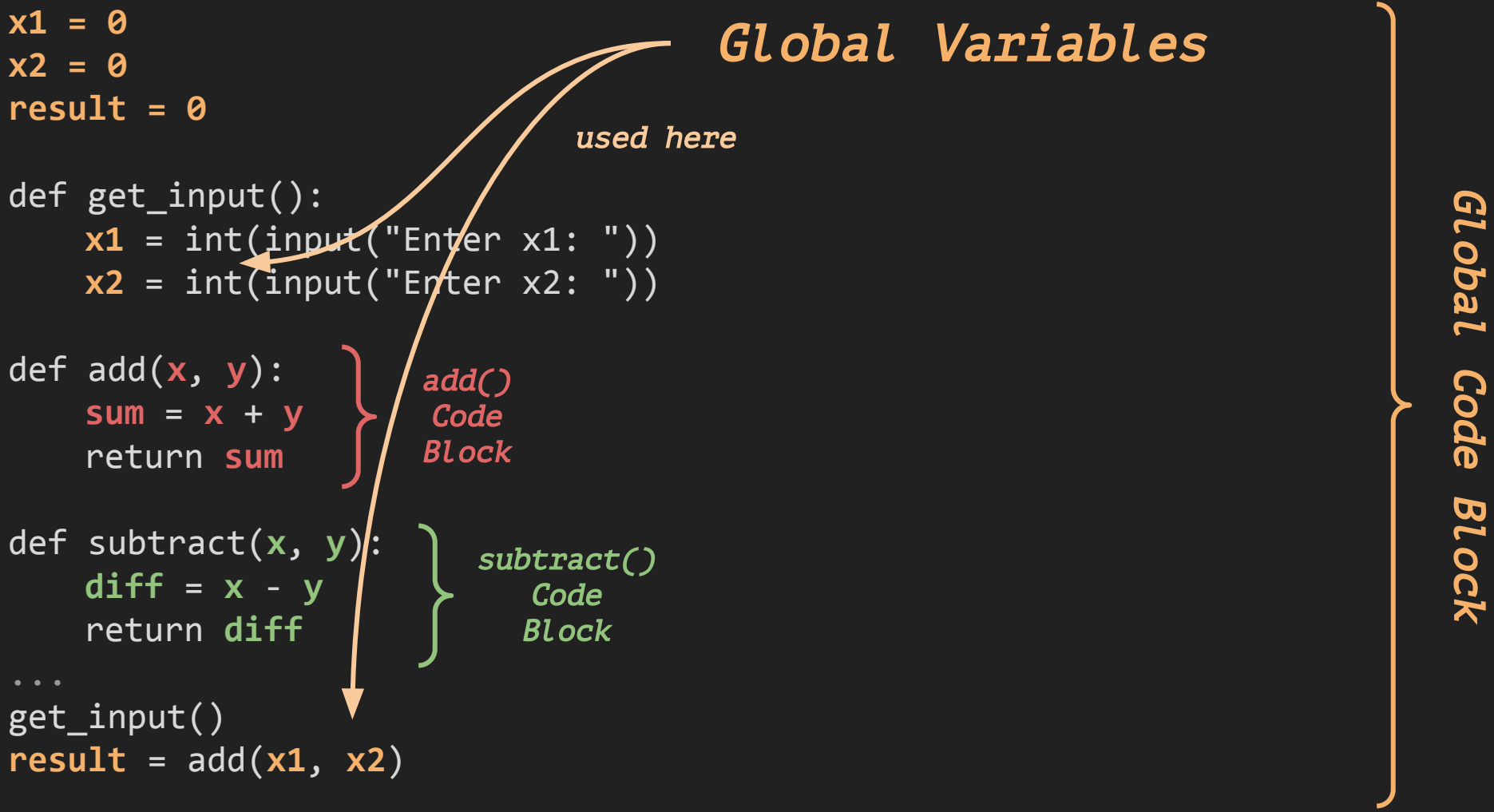
Global Variables

declared here

add() Code Block

subtract() Code Block

Global Code Block


x1 = 0
x2 = 0
result = 0
Global Variables
used here
def get_input():
x1 = int(input("Enter x1: "))
x2 = int(input("Enter x2: "))
def add(x, y):
sum = x + y
return sum
add()
Code
Block
def subtract(x, y):
diff = x - y
return diff
subtract()
Code
Block
...
get_input()
result = add(x1, x2)
Global Code Block

```
x1 = 0
x2 = 0
result = 0

def get_input():
    x1 = int(input("Enter x1: "))
    x2 = int(input("Enter x2: "))

def add(x, y):
    sum = x + y
    return sum

def subtract(x, y):
    diff = x - y
    return diff
...
get_input()
result = add(x1, x2)
```

add() Code Block

Global Variables

***Local Variables** - visible by instructions of add function*

Global Code Block

```
x1 = 0
x2 = 0
result = 0

def get_input():
    x1 = int(input("Enter x1: "))
    x2 = int(input("Enter x2: "))

def add(x, y):
    sum = x + y
    return sum

def subtract(x, y):
    diff = x - y
    return diff
...
get_input()
result = add(x1, x2)
```

Global Variables

Global Code Block

add() Code Block

Local Variables - visible by instructions of add function

subtract() Code Block

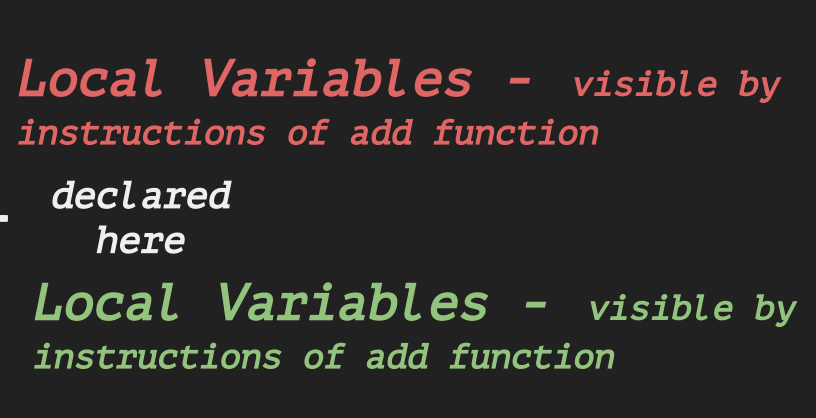
Local Variables - visible by instructions of add function

```
x1 = 0
x2 = 0
result = 0

def get_input():
    x1 = int(input("Enter x1: "))
    x2 = int(input("Enter x2: "))

def add(x, y):
    sum = x + y
    return sum

def subtract(x, y):
    diff = x - y
    return diff
...
get_input()
result = add(x1, x2)
```



Global Variables

Local Variables - *visible by instructions of add function*

declared here

Local Variables - *visible by instructions of add function*

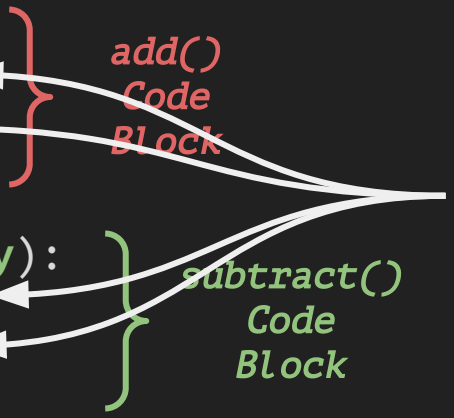
Global Code Block

```
x1 = 0
x2 = 0
result = 0

def get_input():
    x1 = int(input("Enter x1: "))
    x2 = int(input("Enter x2: "))

def add(x, y):
    sum = x + y
    return sum

def subtract(x, y):
    diff = x - y
    return diff
...
get_input()
result = add(x1, x2)
```



Global Variables

Local Variables - *visible by instructions of add function*

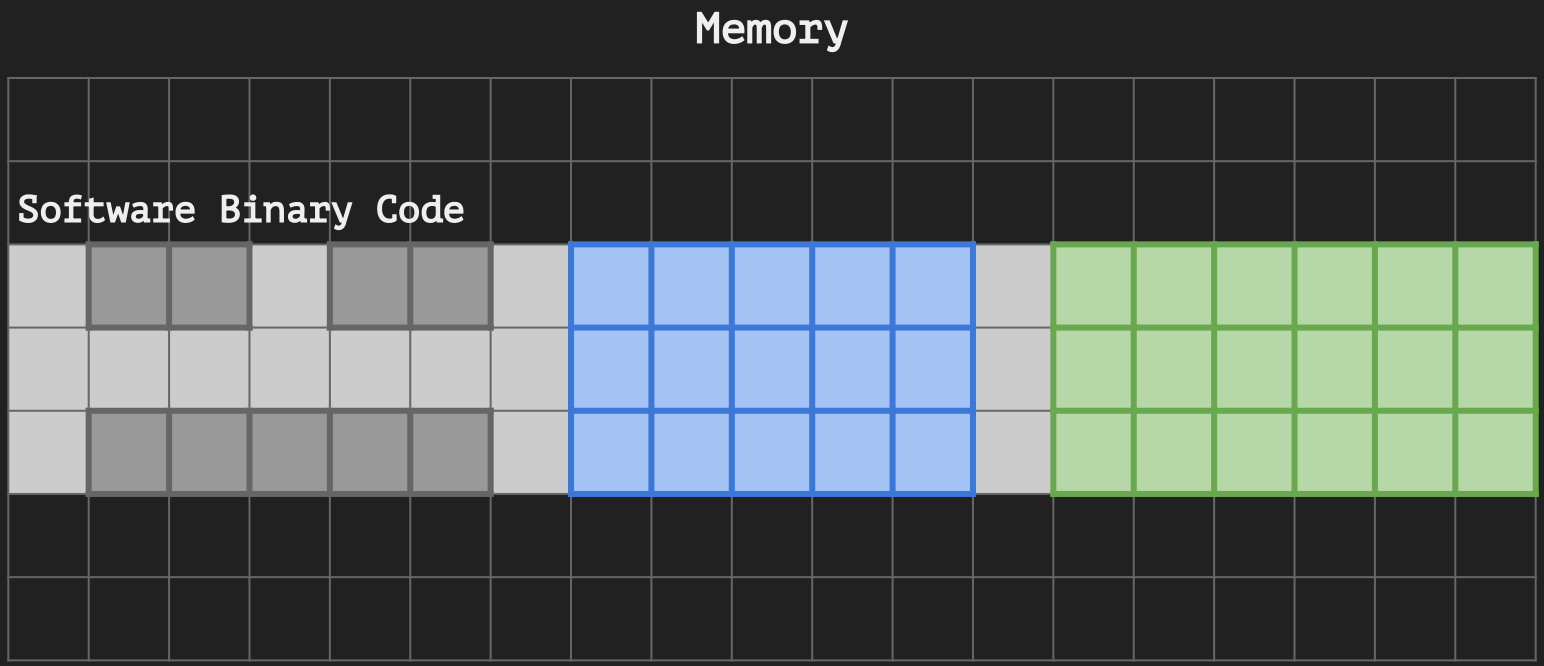
Local Variables - *visible by instructions of add function*

Global Code Block

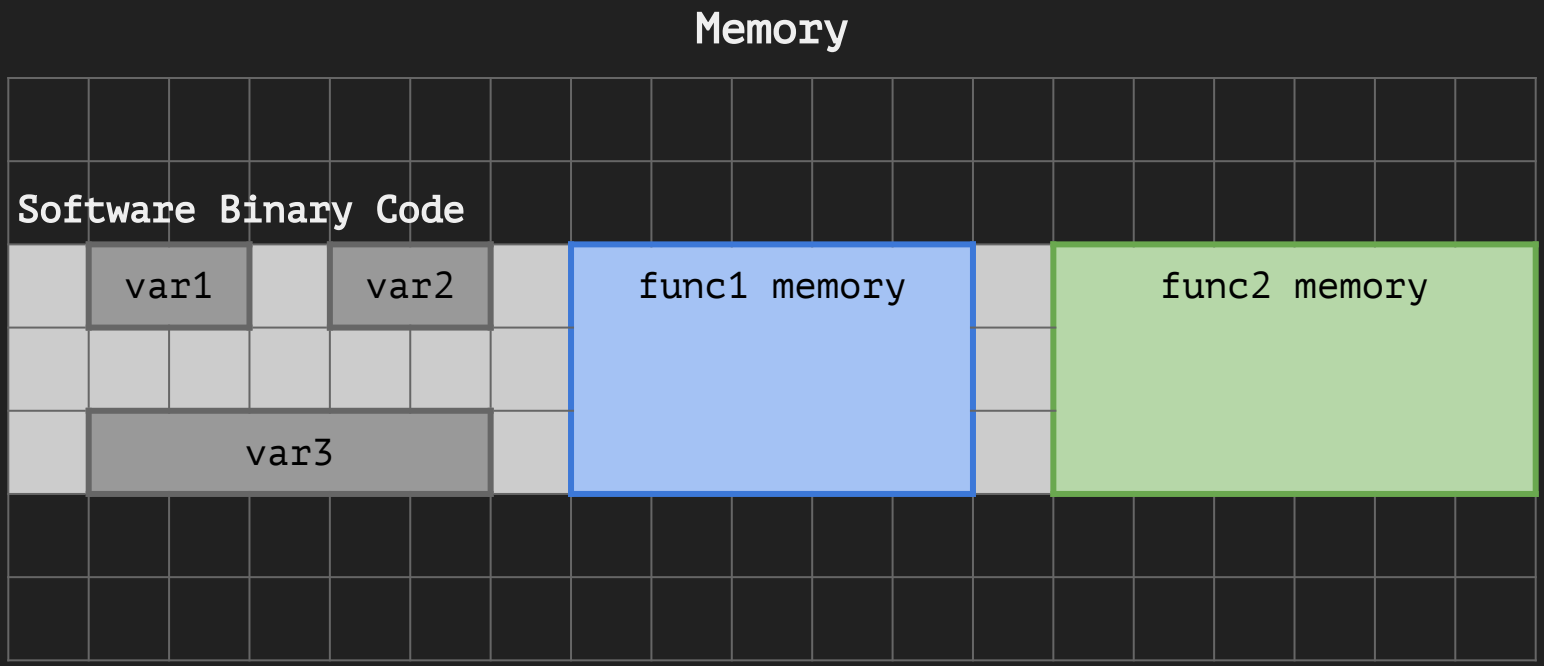
How does code work?

Variables in Action

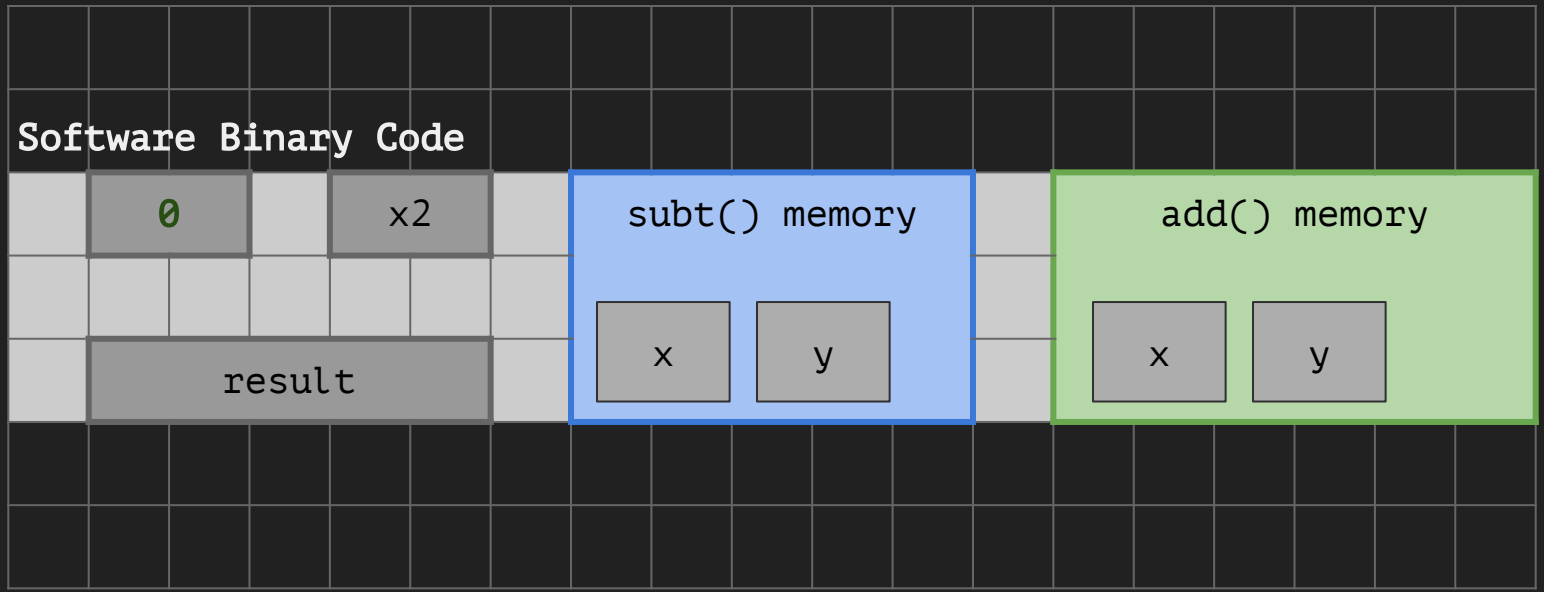
Variables: *are translated into their binary encoding and added to the binary code of the software.*



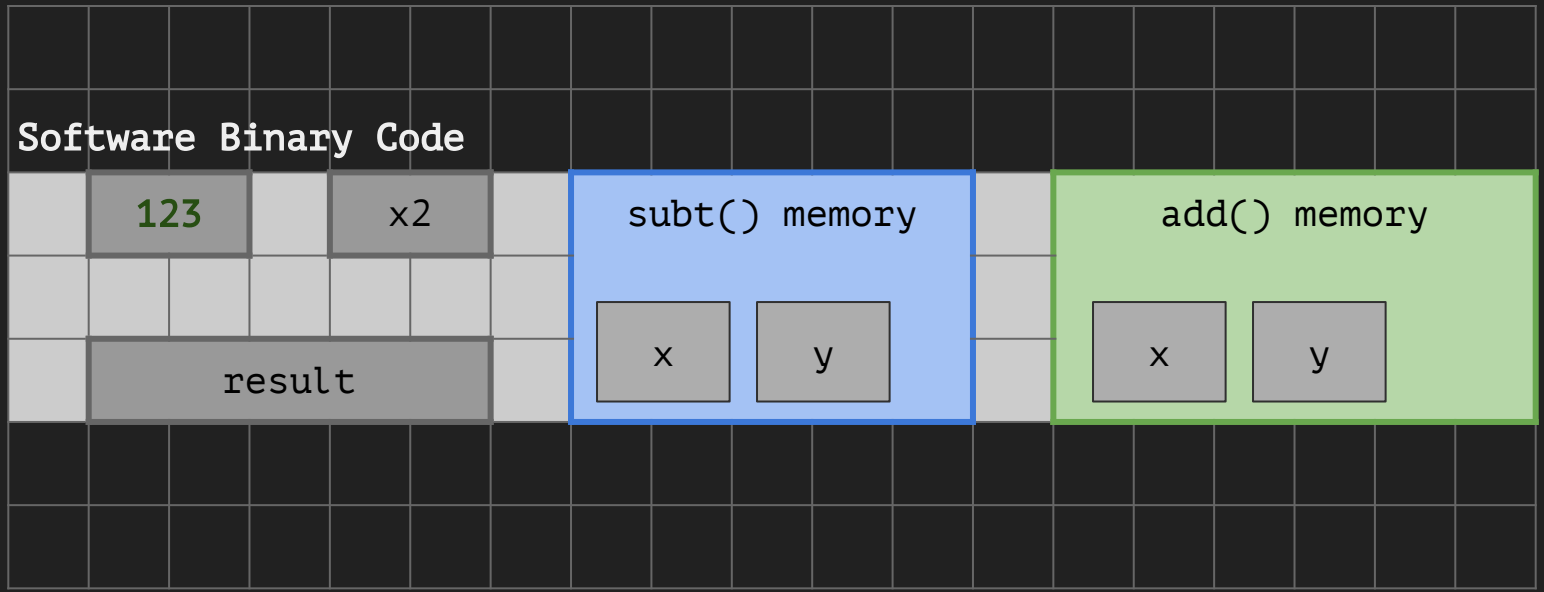
Variables: *values change in the memory locations allocated for the variables*



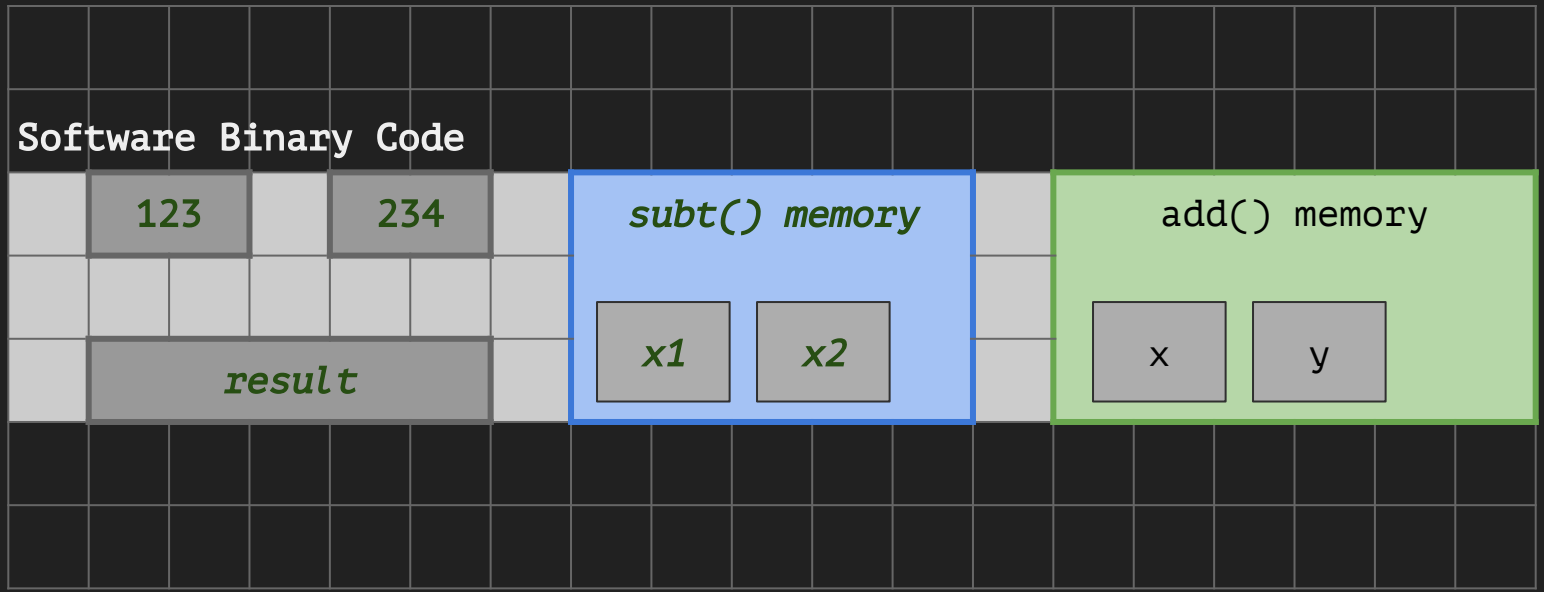
```
x1 = 0
...
def get_input():
    x1 = int(input())
    ...
```



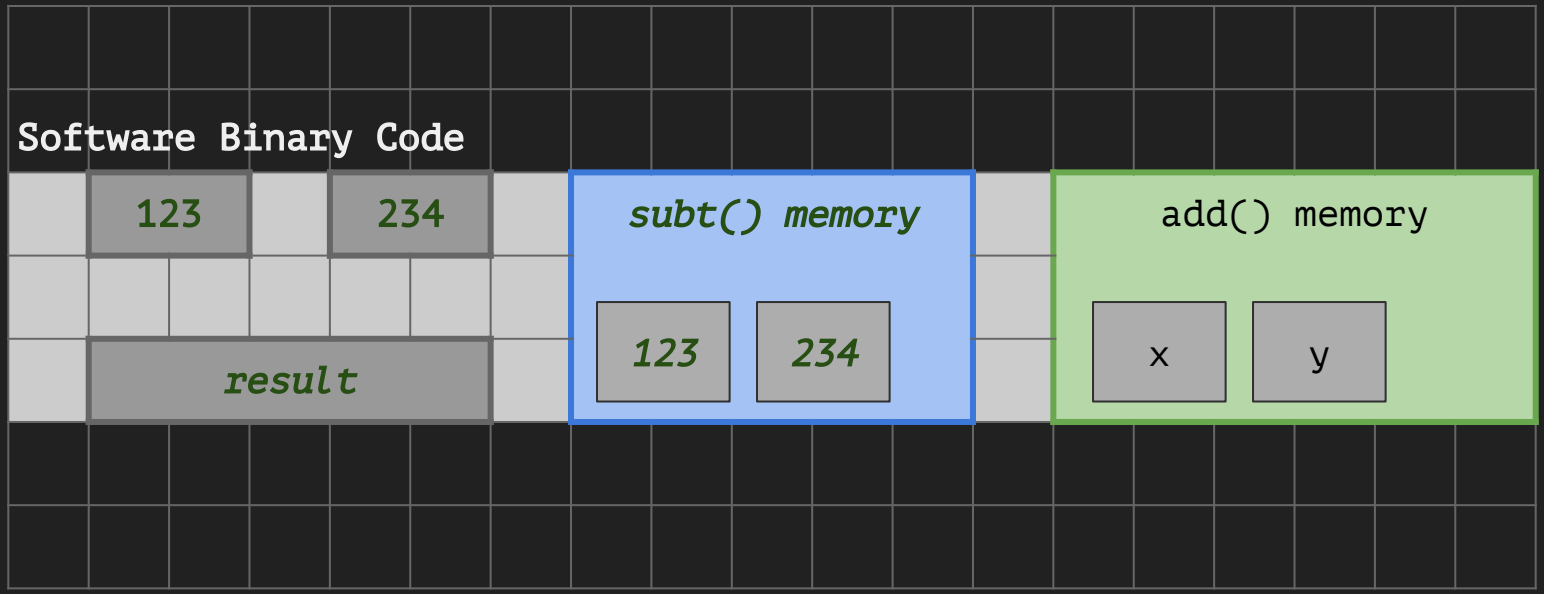
```
x1 = 0
...
def get_input():
    x1 = int(input())
    ...
```



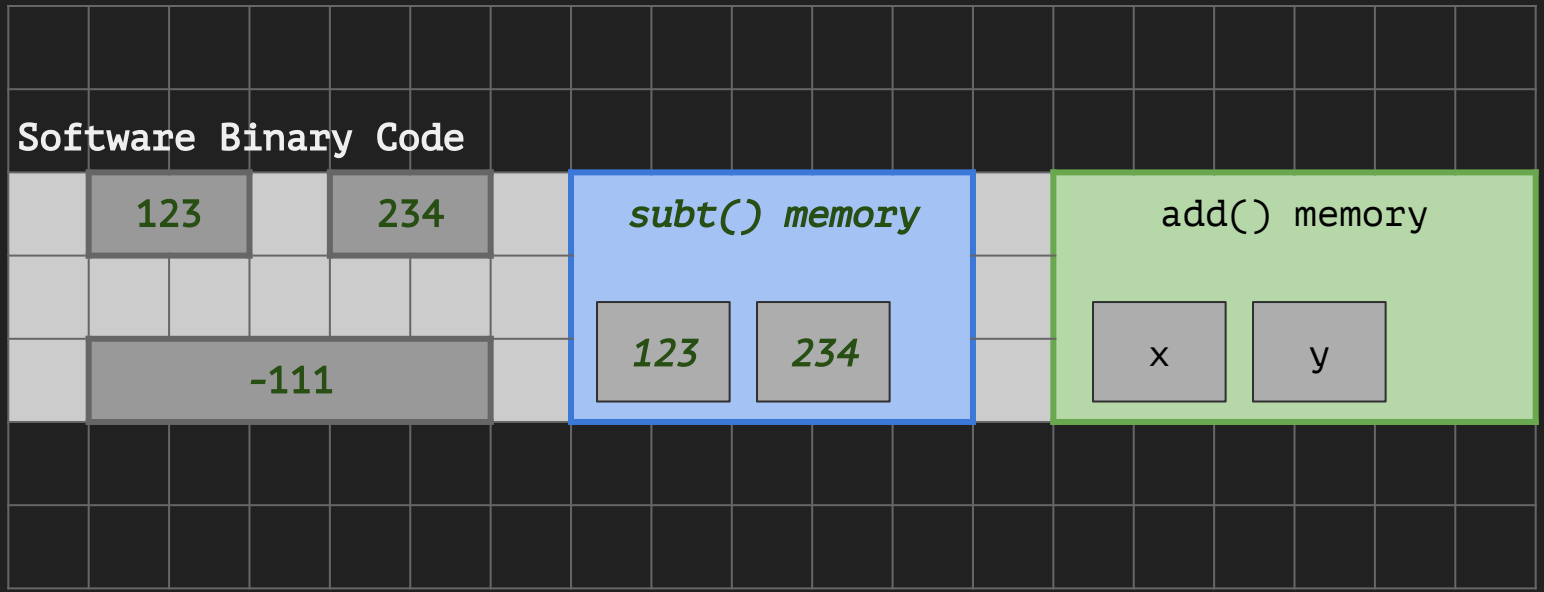
```
...
result = subt(x1, x2)
def subt(x, y):
    return x - y
    ...
```



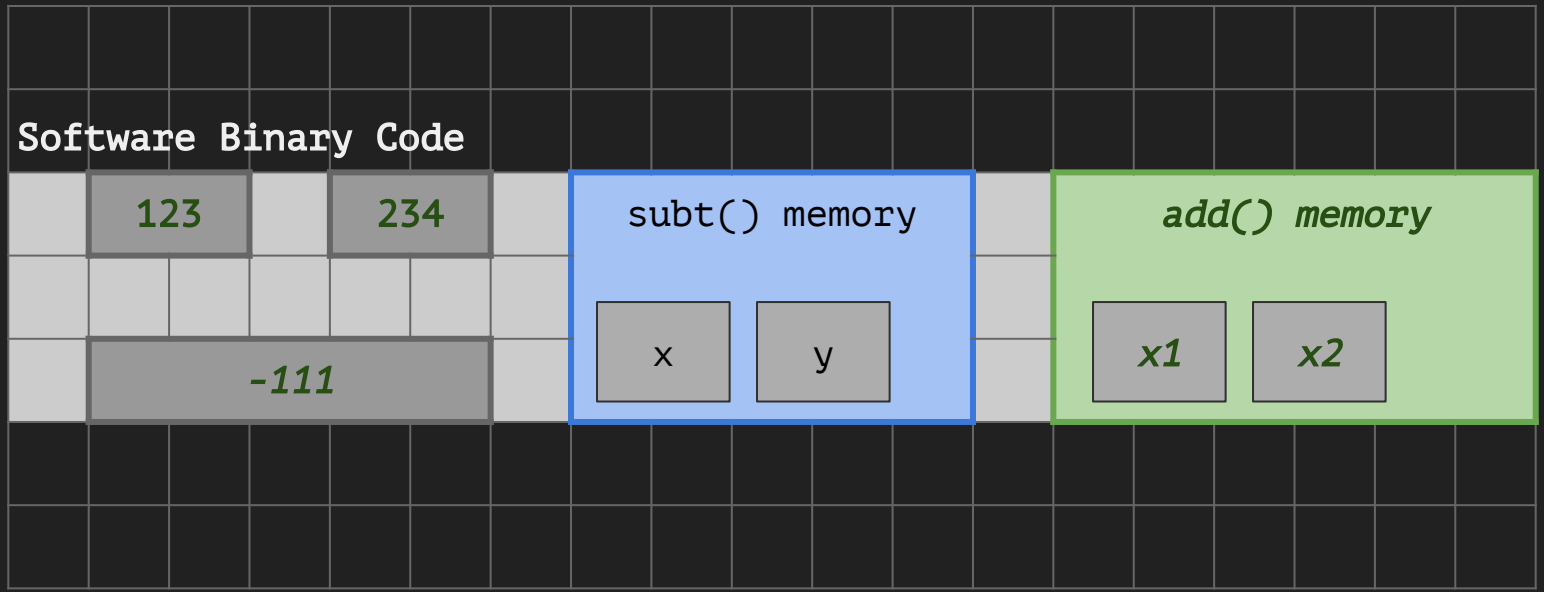
```
...
result = subt(x1, x2)
def subt(x, y):
    return x - y
    ...
```



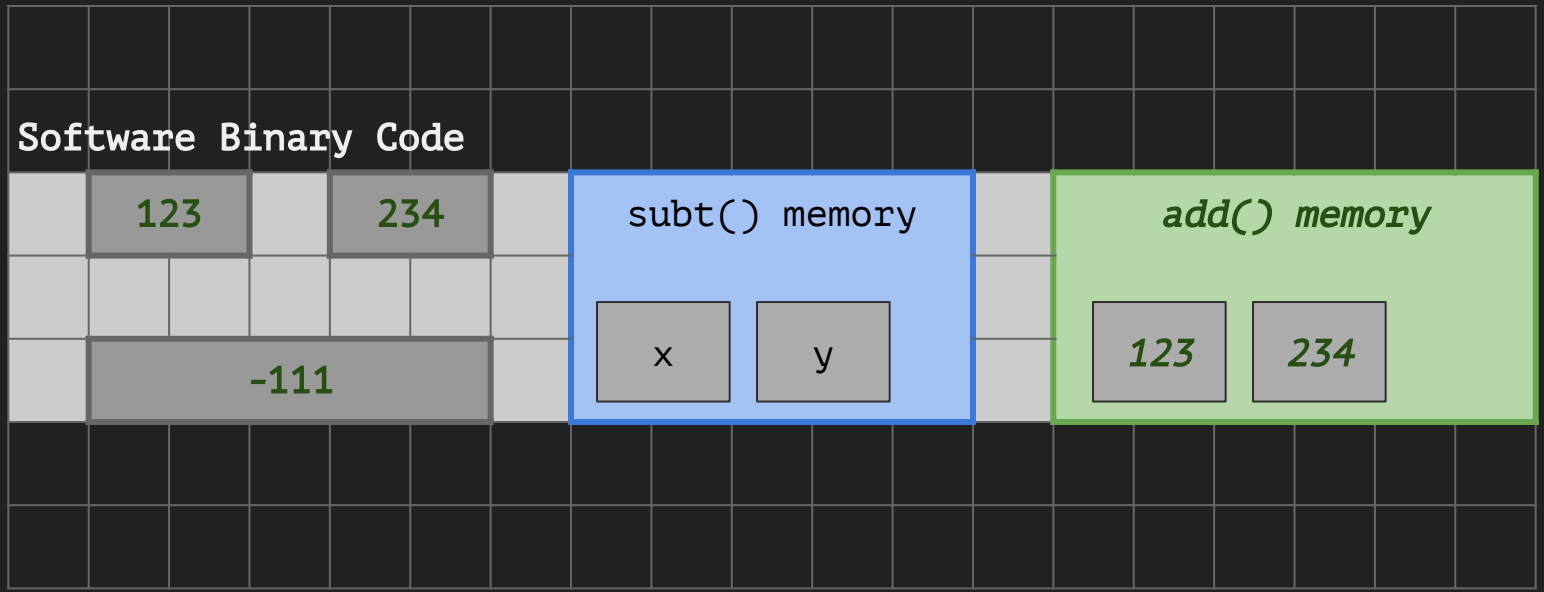
```
...
result = subt(x1, x2)
def subt(x, y):
    return x - y
    ...
```



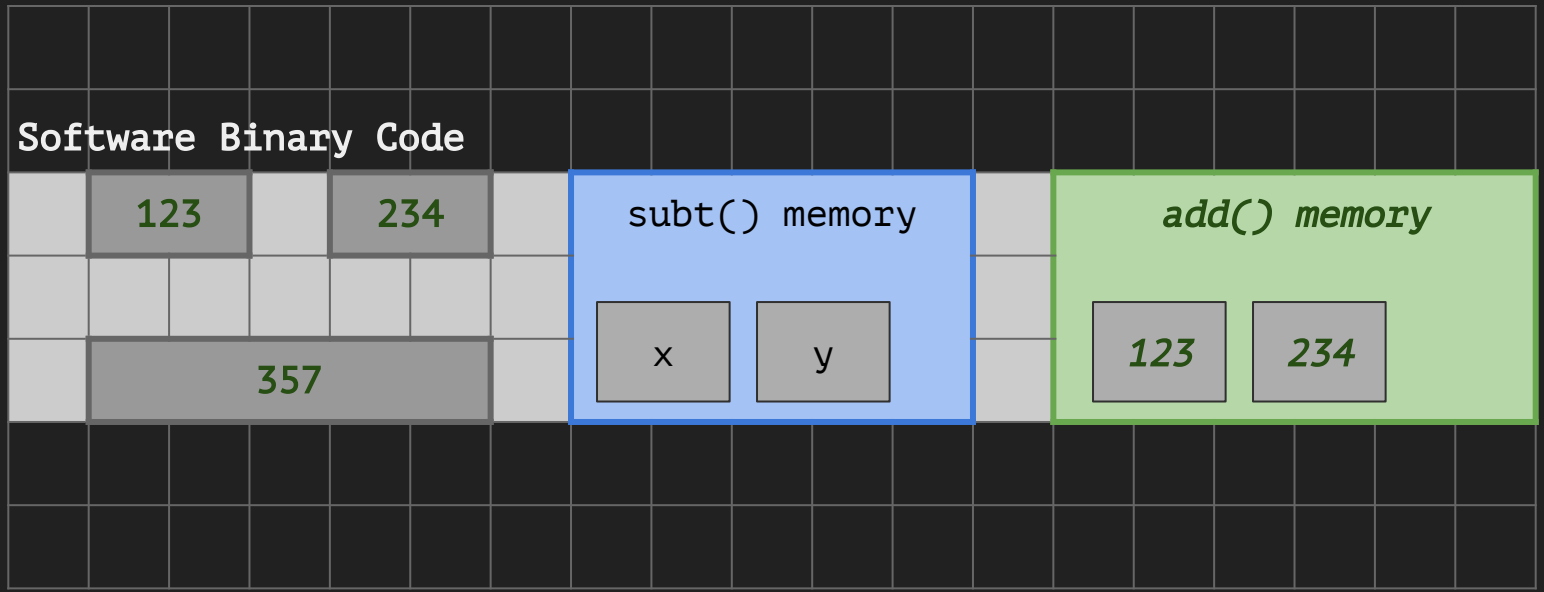
```
...
result = add(x1, x2)
def add(x, y):
    return x + y
    ...
```



```
...
result = add(x1, x2)
def add(x, y):
    return x + y
    ...
```



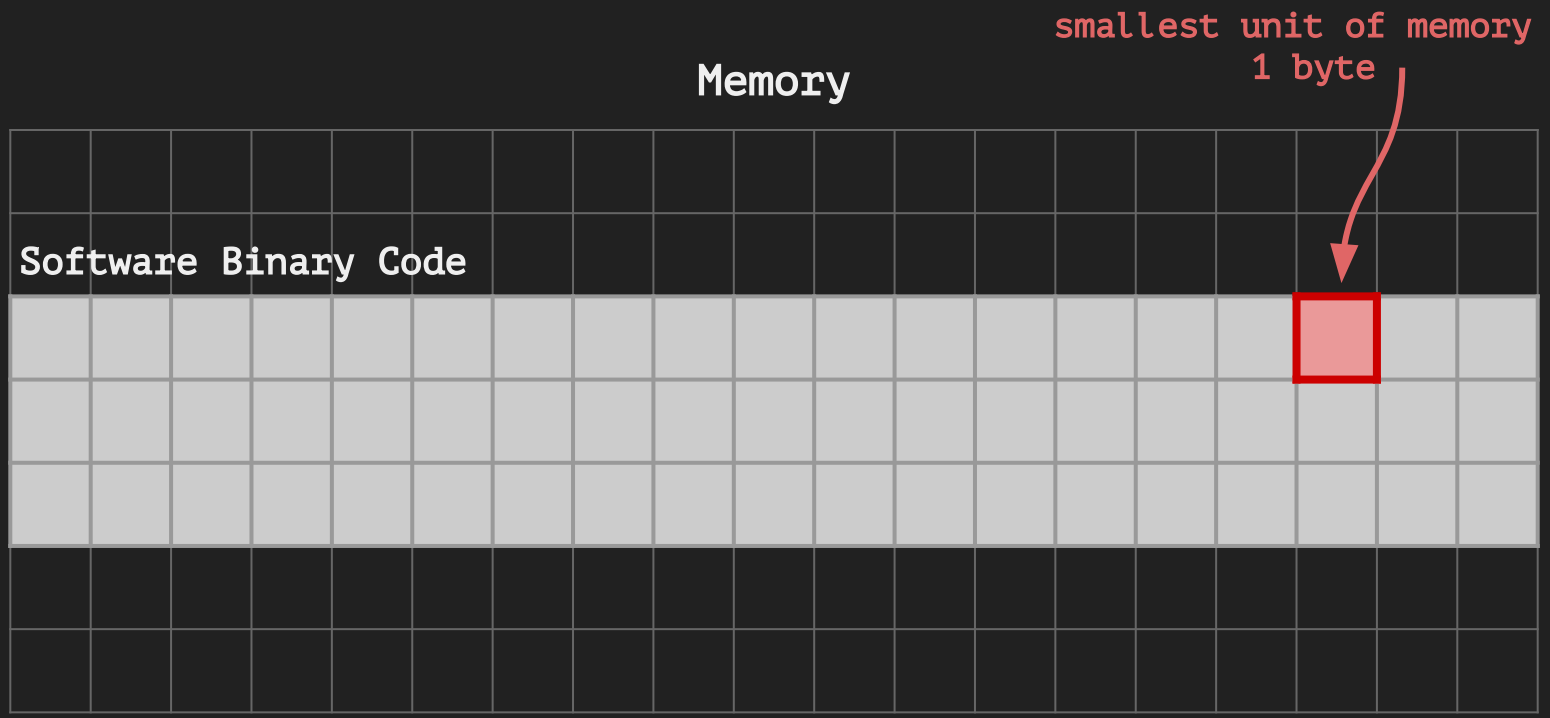
```
...
result = add(x1, x2)
def add(x, y):
    return x + y
    ...
```



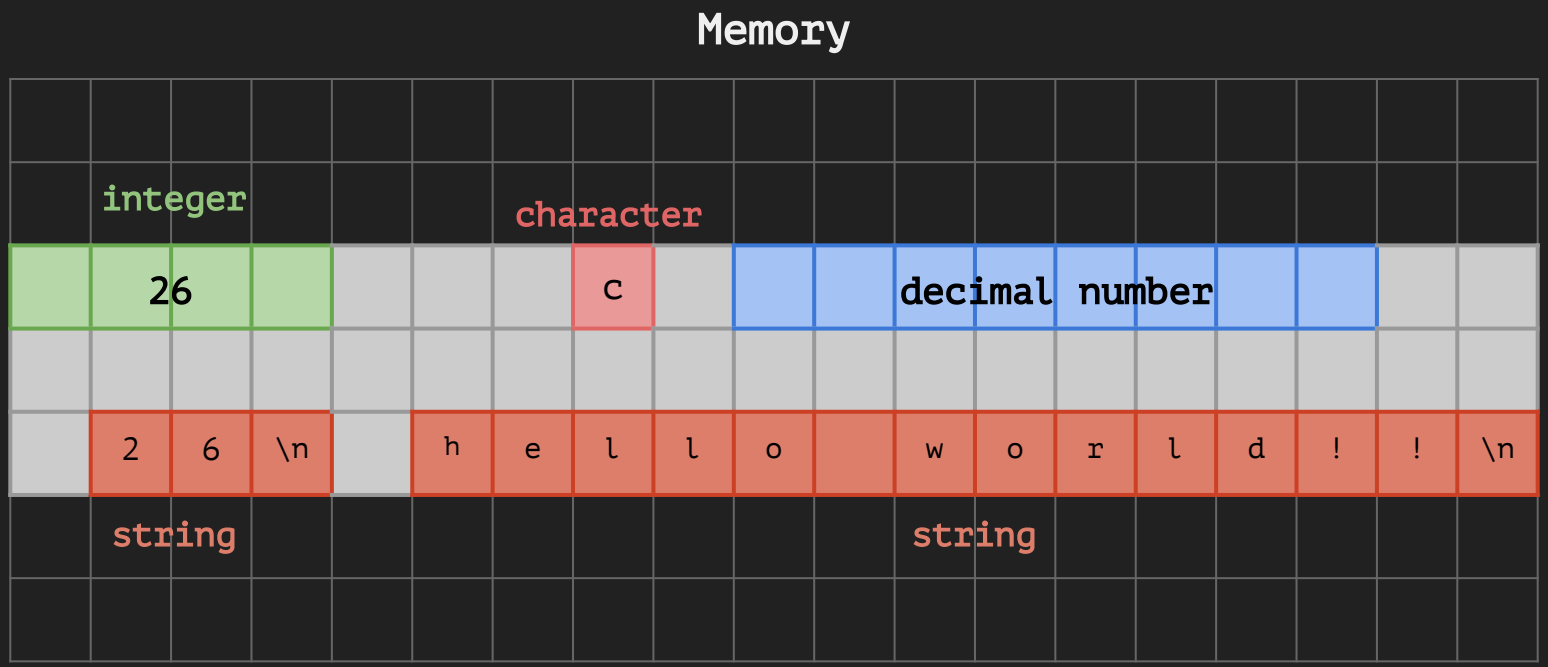
How does code work?

Variable Types

Variables: *consume different amounts of memory based on their data type*



Variables: *consume different amounts of memory based on their data type*



Variables: *have different binary encodings depending on their data type*