

# APS105

# Winter 2012

Jonathan Deber  
jdeber -at- cs -dot- toronto -dot- edu

Lecture 11  
February 8, 2012

# Today

- Function declarations
- Parameters
- Scope
- Assignment 2

# Using Functions

double average(double a, double b)

int getSpeed(void)

void printSquared(int n)

void printHello(void)

# Side Effects

- void functions need a side effect
- Functions with return types may have side effects
- This means you will sometime (intentionally) ignore a return value

numChars: 21

```
int numChars = printf("I have a side effect.");
```

I have a side effect.

```
printf("%d", printf("I have a side effect."));
```

I have a side effect.21

# Functions Without Parameters

- Do the same thing each time
- (Unless they read data from outside world)

```
void printHello(void)
{
    printf("Hello.\n");
}

printHello();    Hello.
printHello();    Hello.
int i = printHello(); Error
printHello;      Legal, but not what you expect
```

# Function Declarations

```
int main(void)
{
    printHello();

    return 0;
}
```

```
void printHello(void)
{
    printf("Hello.\n");
}
```

warning: conflicting types  
for 'printHello'

warning: previous implicit  
declaration of 'printHello'  
was here

# Function Declarations

- Functions need to be declared before use

```
int main(void)
{
    printHello();

    return 0;
}

void printHello(void)
{
    printf("Hello.\n");
}
```

# Function Declarations

- Functions need to be declared before use

```
void printHello(void)
{
    printf("Hello.\n");
}

int main(void)
{
    printHello();

    return 0;
}
```

# Function Declarations

- Functions need to be declared before use

```
void printHello(void);           int i;  
  
int main(void)  
{  
    printHello();  
  
    return 0;  
}  
  
void printHello(void)           int i = 0;  
{  
    printf("Hello.\n");  
}
```

# Function Declarations

- We call the declaration a “prototype”

```
double average(double a, double b);  
void printSquared(int n);
```

stdio.h contains the function prototypes  
for printf(), scanf(), etc.

# Parameters vs.Arguments

```
double average(double a, double b);  
int main(void)  
{  
    double x = 5.0;  
    double y = 15.0;  
    double avg = average(x, y);  
  
    return 0;  
}  
double average(double a, double b)  
{  
    return (a + b) / 2;  
}
```

The diagram illustrates the mapping of arguments to parameters in a function call. It features a central light-gray rectangular box containing the code snippet. Four arrows point from the labels "arguments" and "parameters" to specific parts of the code: one arrow from "arguments" points to the expression "average(x, y)", while two arrows from "parameters" point to the formal parameters "a" and "b" in the definition of the `average` function.

# Parameters

- Get created and set automatically

```
double average(double a, double b)
{ double a;
a = 5.0; → double b;
b = 15.0; return (a + b) / 2;
}
```

```
double average(double a, double b)
{
    double a,b; Error
    return (a + b) / 2;
}
```

# Array Parameters

```
int sumArray(int a[])
{
    int sum = 0;
    for (int i = 0;           ; i++)
    {
        sum += a[i];
    }
    return sum;
}
```

```
int list[] = {1, 2, 3};
int sum = sumArray(list);
```

# Array Parameters

```
int sumArray(int a[], int n)
{
    int sum = 0;
    for (int i = 0; i < n; i++)
    {
        sum += a[i];
    }
    return sum;
}
```

```
int list[] = {1, 2, 3};
int sum = sumArray(list, 3);
```

# Multi-Dimensional Arrays

- Need to specify the length of every dimension other than the first

`int sumArray(int a[]);` Legal  
(but it might be impossible to implement)

`int sum2DArray (int a[][]);` Error

`int sum2DArray (int a[][][], int n, int m);` Error

`int sum2DArray (int a[][10], int n);`

`int sum2DArray (int a[][N], int n);`

C99 only

`int sum2DArray (int n, int m, int a[n][m]);`

# return

- Functions with a return type must have return *expression*
- Function immediately exits

```
double sum(double x, double y)
{
    double sum = x + y;
    return sum;
    sum += x;    } Never executed
    return sum;
}
```

# return

- Type conversion rules apply

```
int squared(double x)
{
    double squared = x * x;
    return squared;
}
```

```
double result = squared(2.5);
```

6.0

# return and void

- void functions can have a return

```
void printHello(void)
{
    printf("Hello.\n");
    return;
    printf("Goodbye.\n"); Never executed
}
void printHello(void)
{
    printf("Hello.\n");
    return 42; Error
}
```

# Function Naming

- Same rules as variables
- Shares same namespace as variables

```
double sum(double a, double b);
```

```
int main(void)
{
    int sum;    Error
    ...
}
```

# Function Naming

- Doesn't distinguish based on type

```
double sum(double a, double b);
```

```
int sum(double a, double b);
```

Error

```
int sum(int a, int b);
```

Error

```
int isum(int a, int b);
```

```
double fsum(double a, double b);
```

```
int sumArray(int a[], int n)
{
    int result = 0;
    for (int i = 0; i < n; i++)
    {
        result += a[i];
    }
    return result;
}
```

```
int multiplyArray(int a[], int n)
{
    int result = 1;
    for (int i = 0; i < n; i++)
    {
        result *= a[i];
    }
    return result;
}
```

```
int main(void)
{
    int result = 0;
    for (int i = 0; i < n; i++)
    {
        result += a[i];
    }
    int result = 1;    Error
    for (int i = 0; i < n; i++)
    {
        result *= a[i];
    }
    return 0;
}
```

# Scope

- The region(s) of a program where a name is valid
- By default, it's the block where a variable is defined
  - Plus any child blocks
  - In C99, only following the declaration

n: 3

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

```
int n = 3;  
int a[n] = {2, 3, 4};
```

n: 3

```
int result = 1;
```

a: 2 3 4

```
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n: 3

a: 2 3 4

result: 1

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n:

a:

result:

i:

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n:

a:

result:

i:

temp:

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n:

a:

result:

i:

temp:

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n:

a:

result:

i:

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n:

a:

result:

i:

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n:

a:

result:

i:

temp:

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n:

a:

result:

i:

temp:

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n:

a:

result:

i:

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n: 3

a: 2 3 4

result: 6

i: 2

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n: 3

a: 2 3 4

result: 6

i: 2

temp: 4

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n: 3

a: 2 3 4

result: 24

i: 2

temp: 4

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n: 3

a: 2 3 4

result: 24

i: 2

```
int n = 3;  
int a[n] = {2, 3, 4};  
  
int result = 1;  
  
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

n: 3

a: 2 3 4

result: 24

i: 3

```
int n = 3;  
int a[n] = {2, 3, 4};
```

n: 3

```
int result = 1;
```

a: 2 3 4

result: 24

```
for (int i = 0; i < n; i++)  
{  
    int temp = a[i];  
    result *= a[i];  
}
```

```
int n = 3;
int a[n] = {2, 3, 4};

int result = 1;

for (int i = 0; i < n; i++)
{
    int temp = a[i];
    result *= a[i];
}
```

```
int main(void)
{
    int n = 3;
    int a[n] = {2, 3, 4};

    int result = 1;

    for (int i = 0; i < n; i++)
    {
        int temp = a[i];
        result *= a[i];
    }

    return 0;
}
```

main()
n: 3      result: 1
a: 2 3 4

```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}

double square(double a)
{
    double result = a * a;
    return result;
}

void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}
```

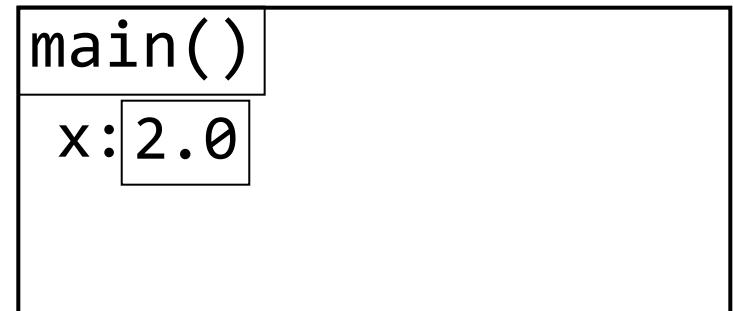
```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}
```

main()

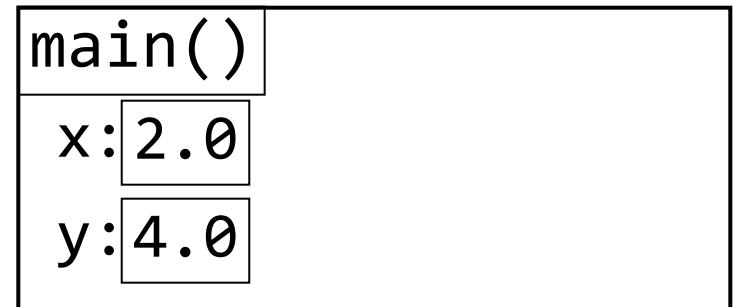
```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}
```



```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

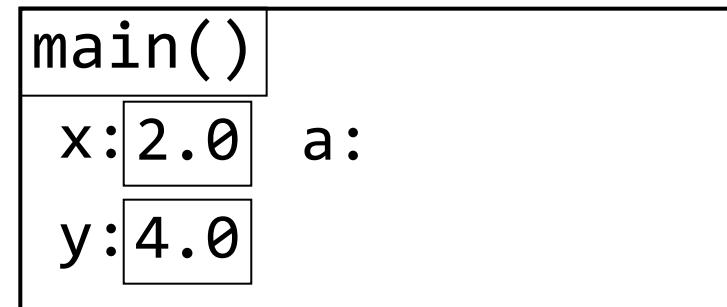
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

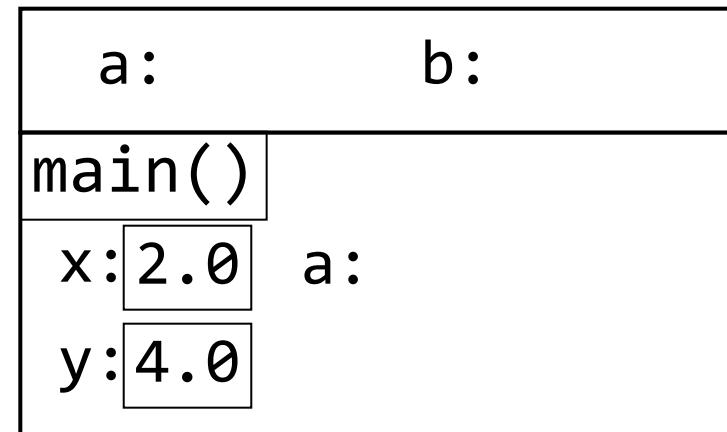
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

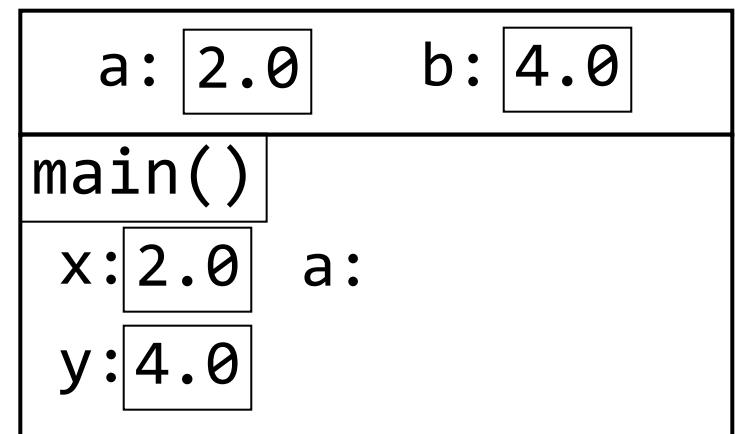
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

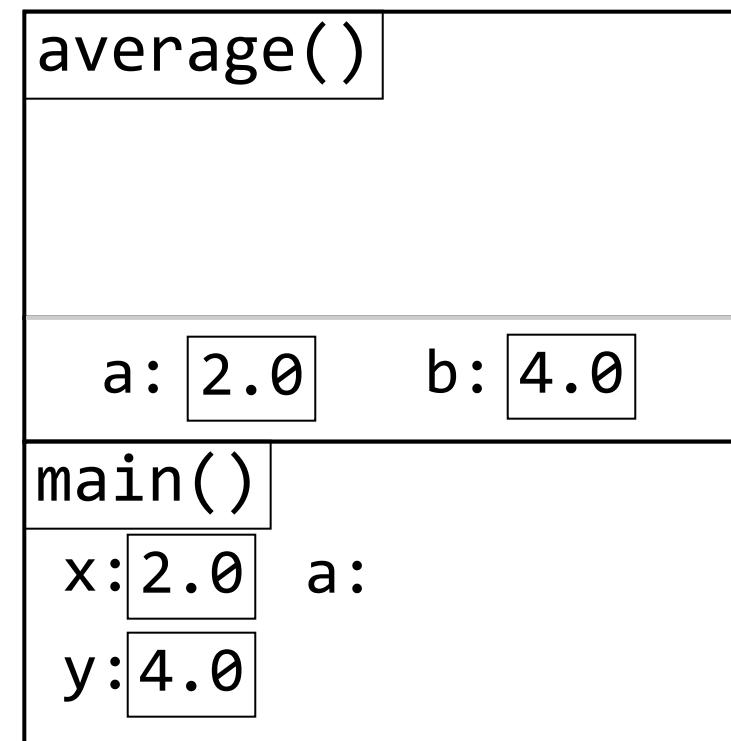
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

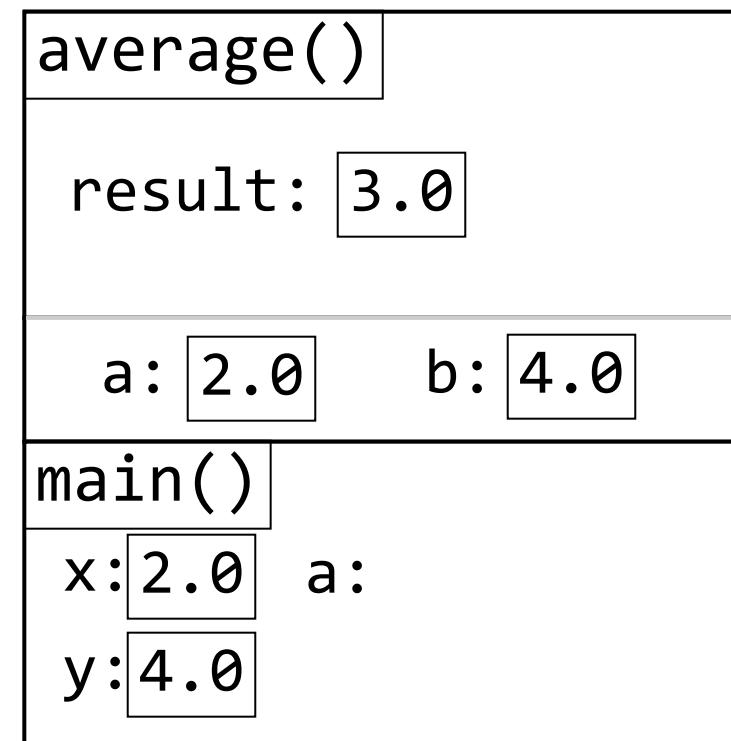
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

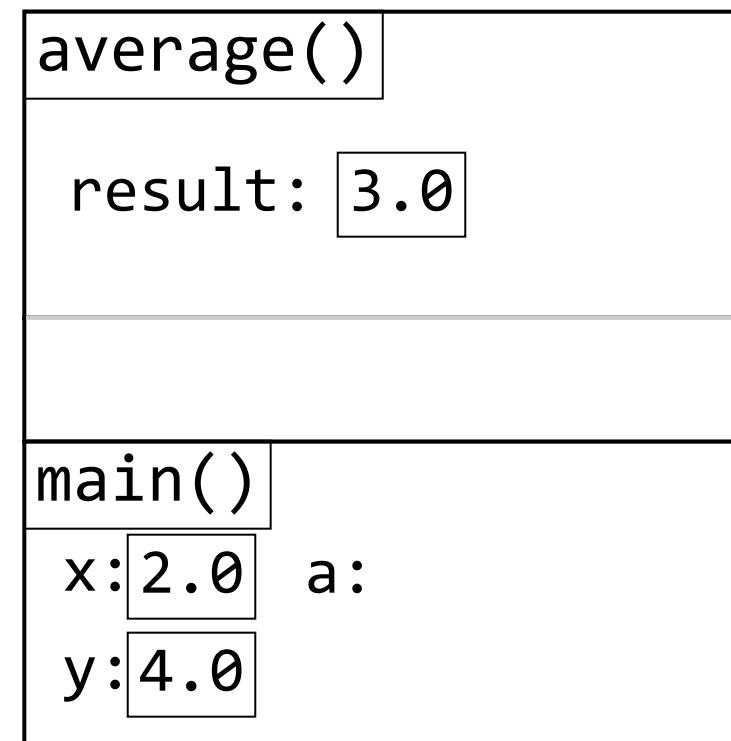
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

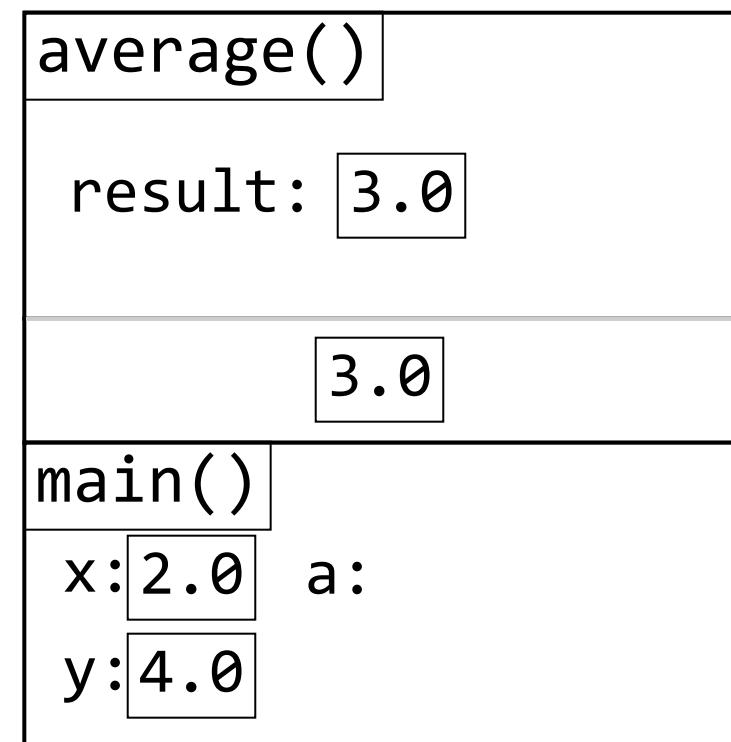
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

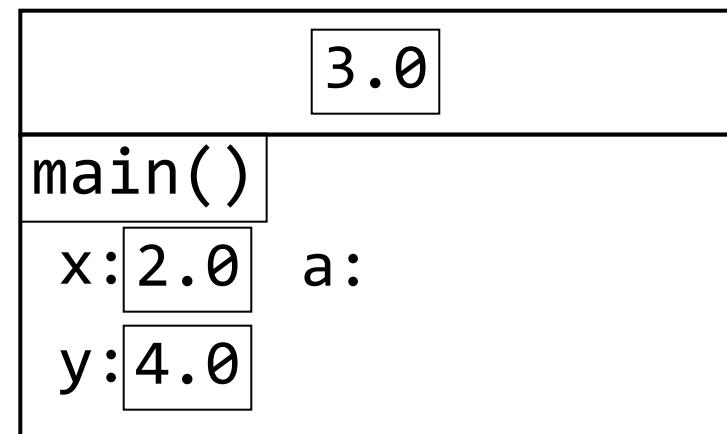
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

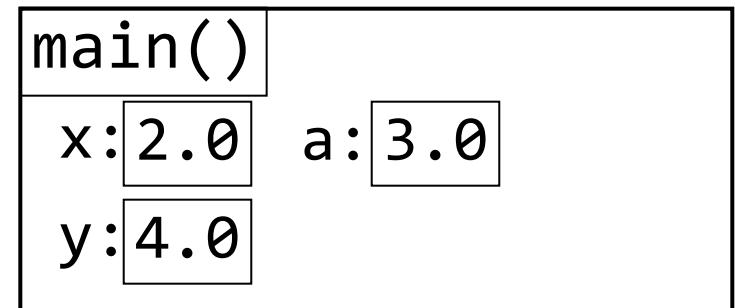
    return 0;
}
```



```
double average(double a, double b)
{
    double result = (a + b) / 2.0;
    return result;
}
```

```
int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}
```



```

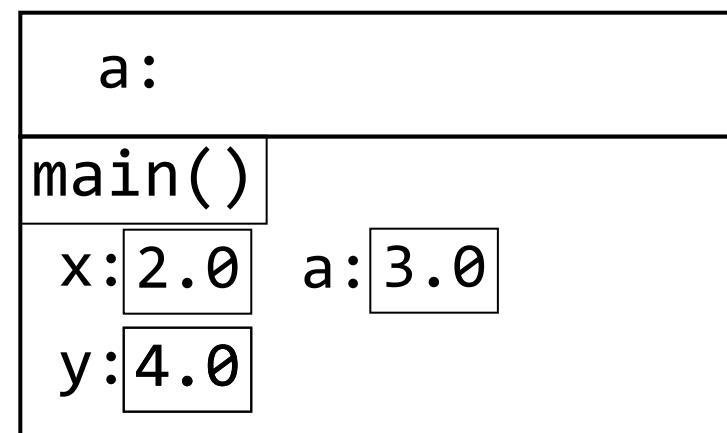
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

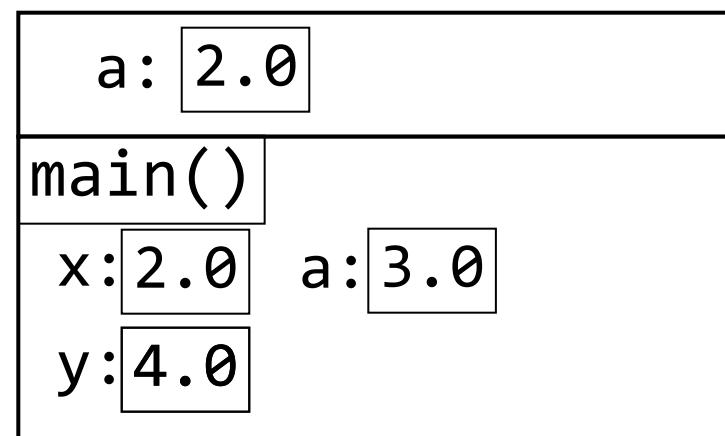
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

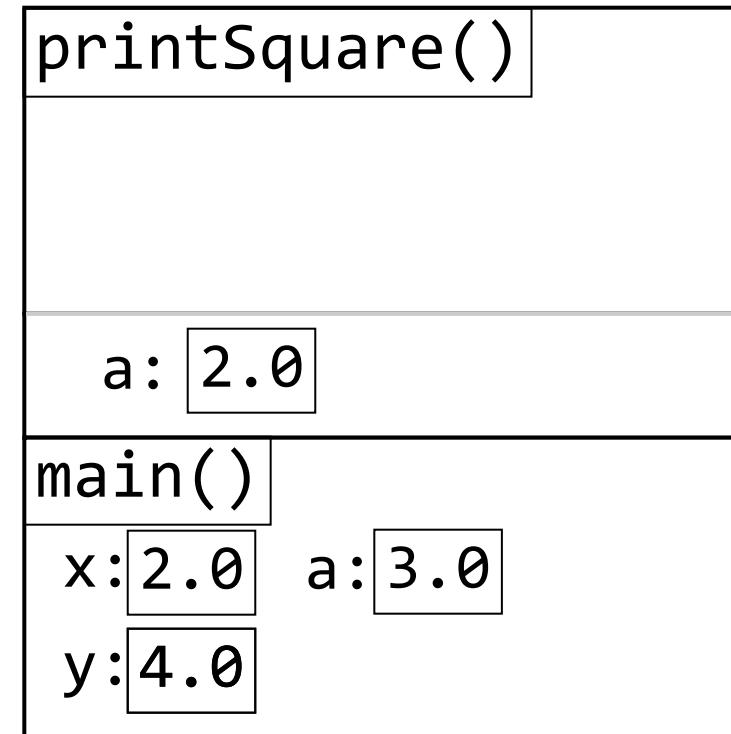
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

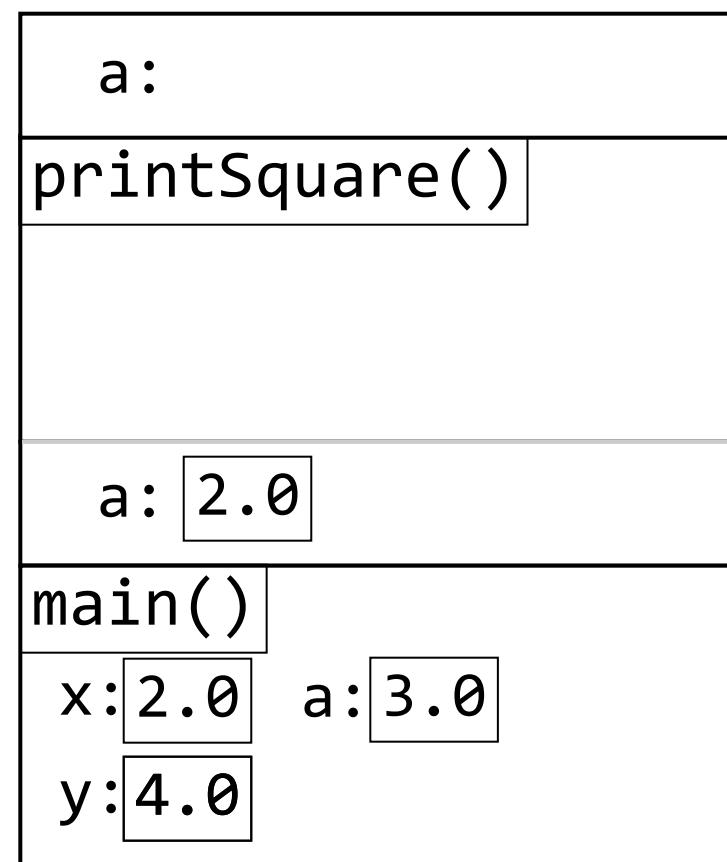
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

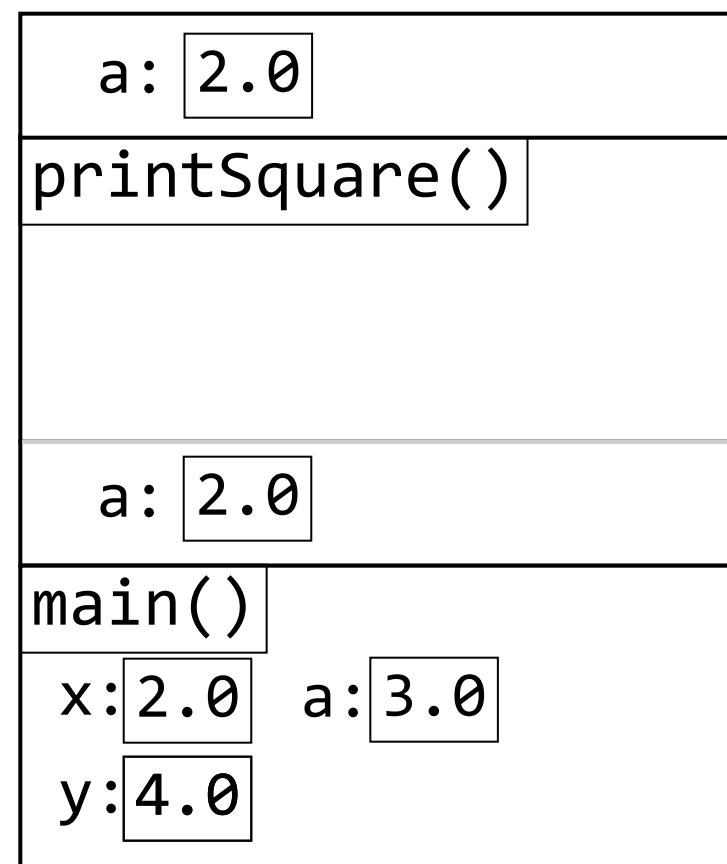
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```

square()

a: 2.0

printSquare()

a: 2.0

main()

x: 2.0 a: 3.0

y: 4.0

```

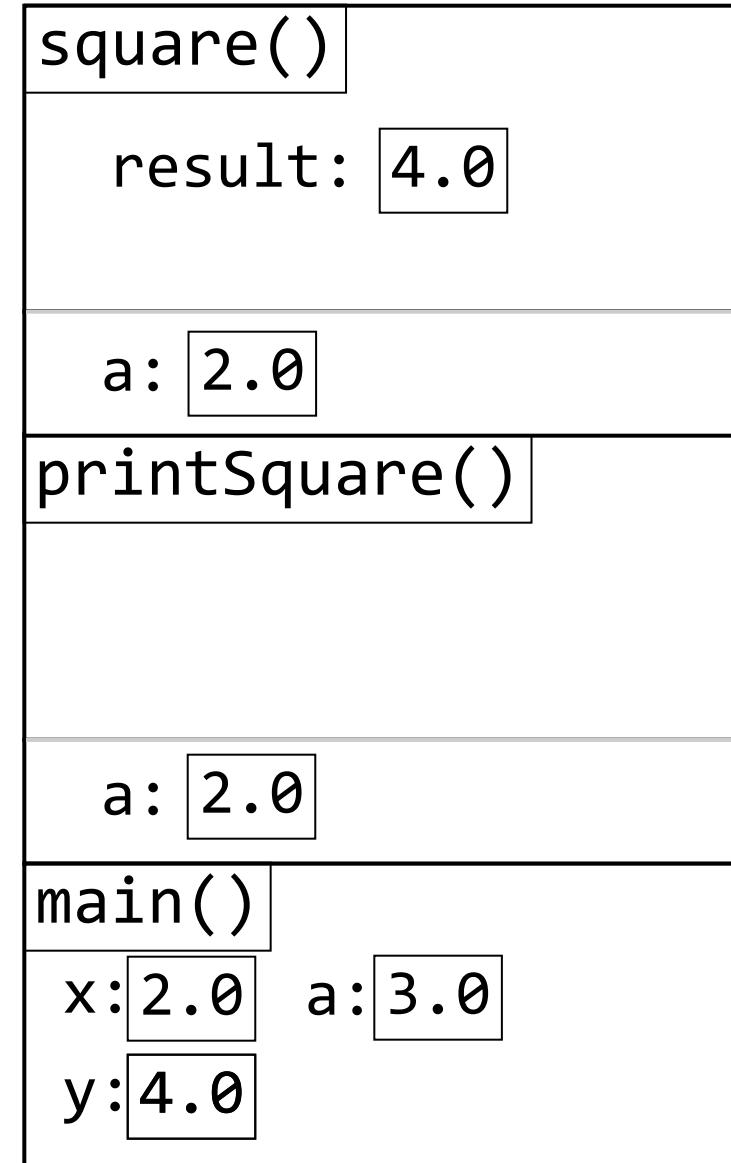
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

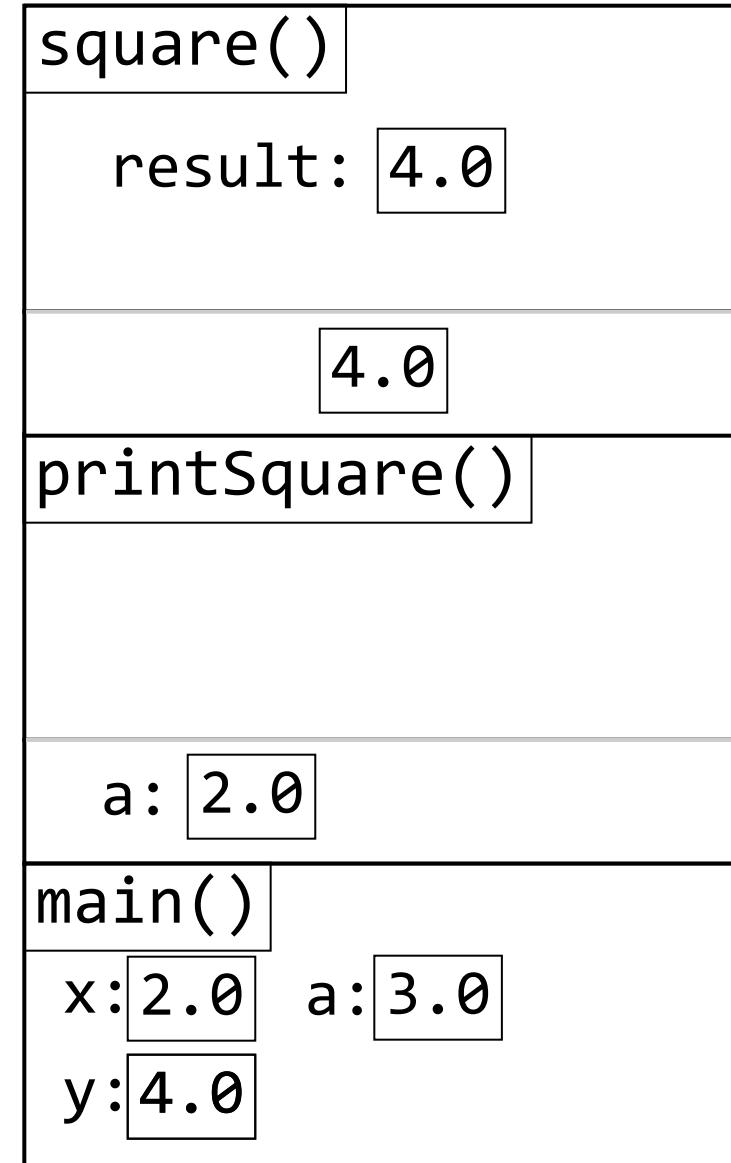
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

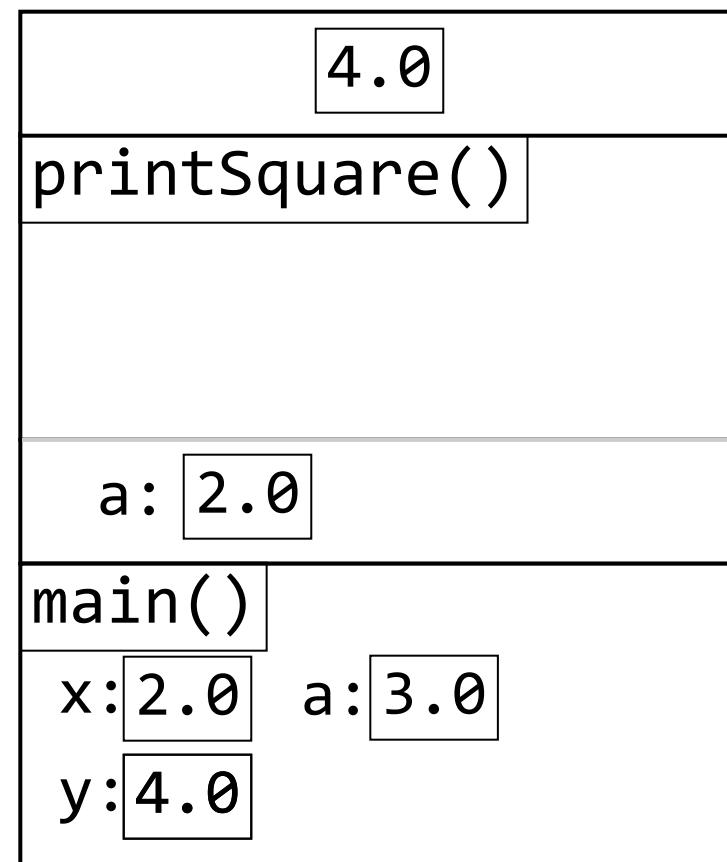
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

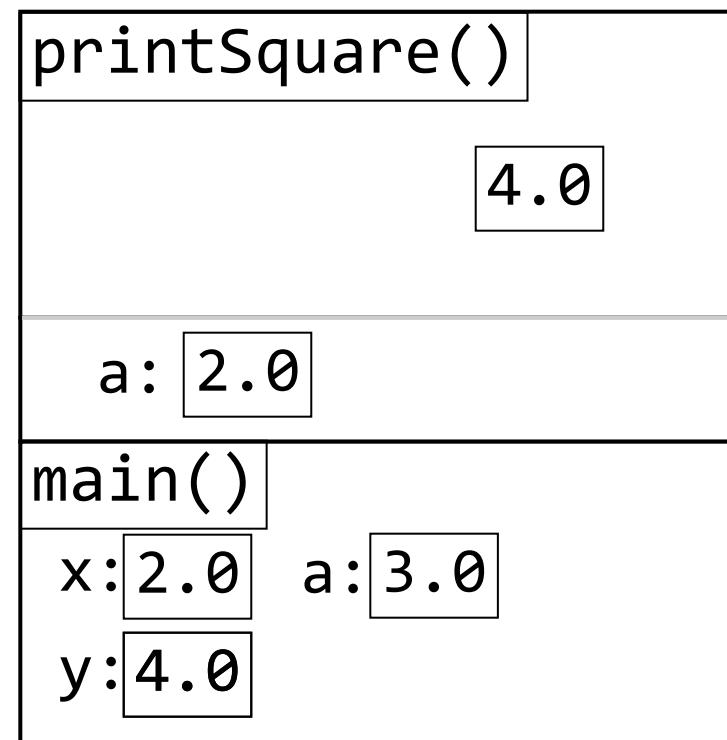
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

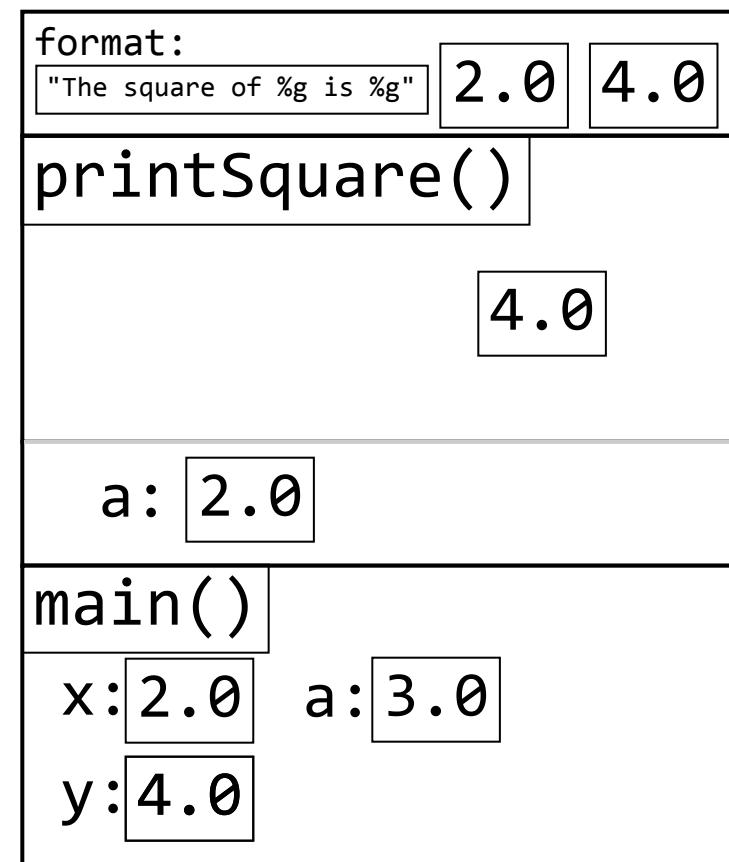
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

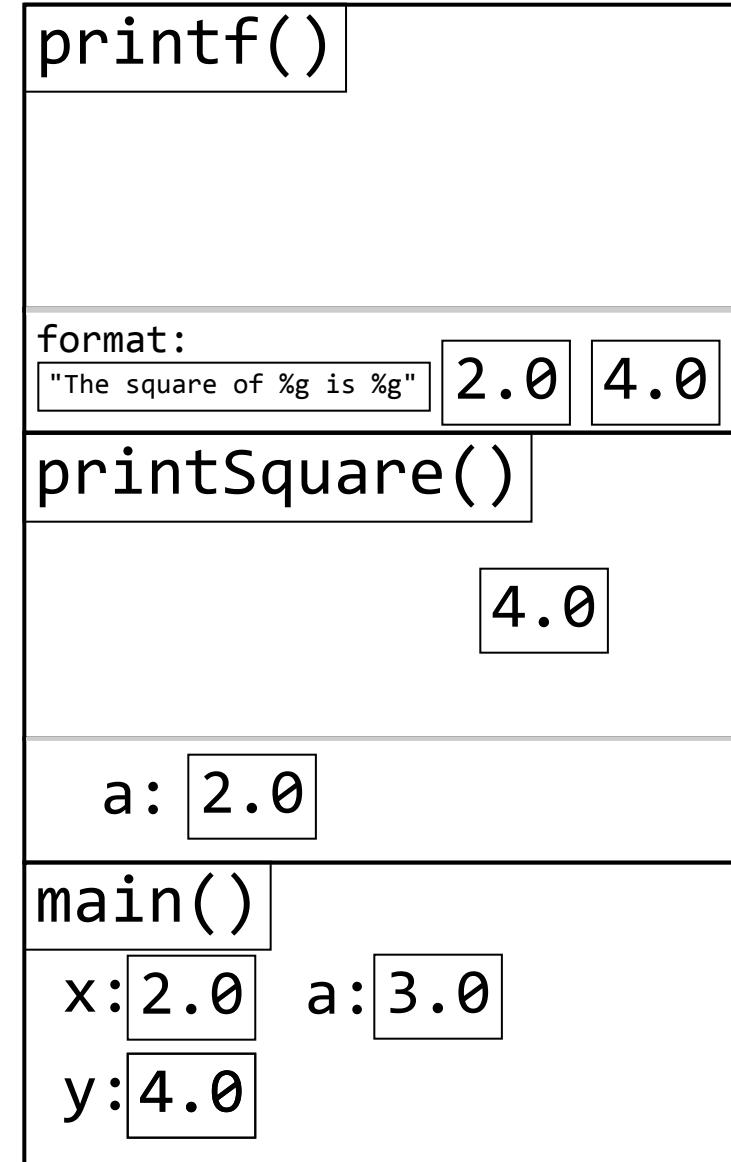
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```



```

void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```

The square of 2 is 4

printf()

format:

"The square of %g is %g"

2.0

4.0

printSquare()

4.0

a: 2.0

main()

x: 2.0

a: 3.0

y: 4.0

```
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}
```

The square of 2 is 4

printf()

printSquare()

4.0

a: 2.0

main()

x: 2.0 a: 3.0

y: 4.0

```
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}
```

The square of 2 is 4

printf()

20

printSquare()

4.0

a: 2.0

main()

x: 2.0 a: 3.0

y: 4.0

```

void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

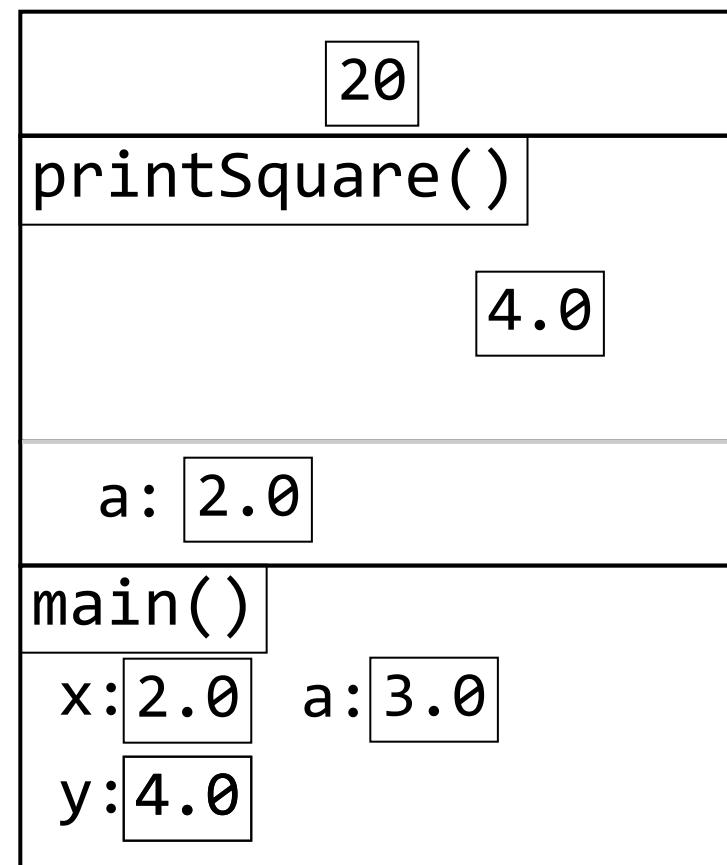
double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```

The square of 2 is 4



```

void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

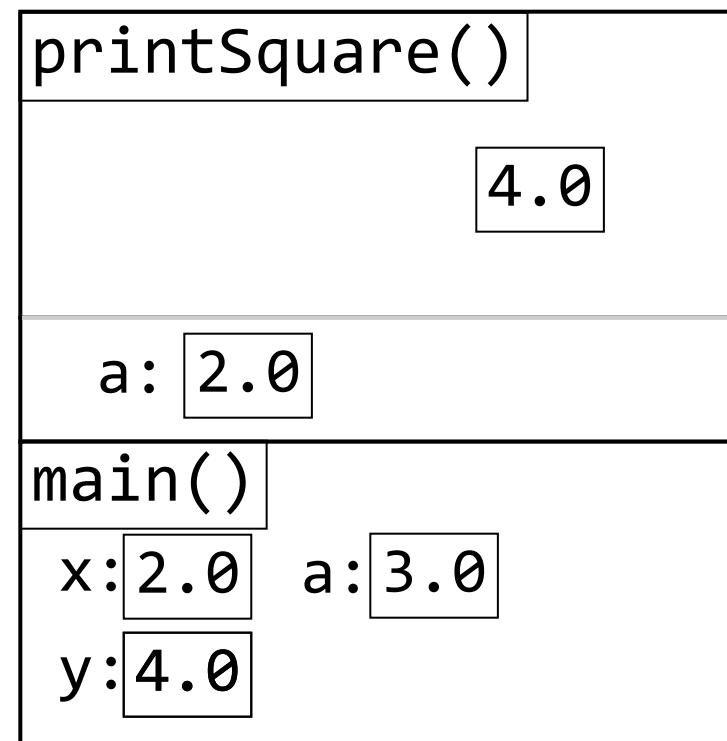
double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```

The square of 2 is 4



```

void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

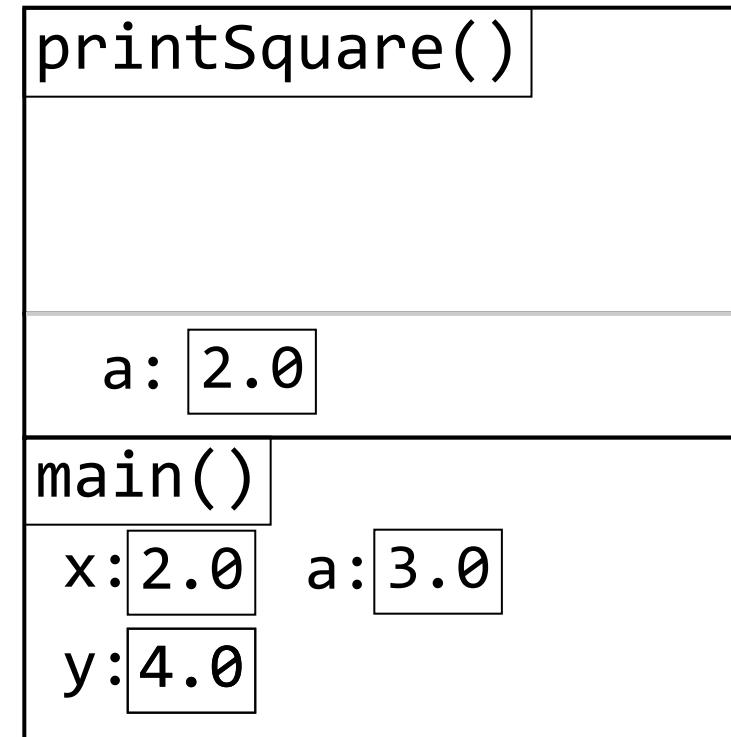
double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```

The square of 2 is 4



```
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}
```

The square of 2 is 4

printSquare()

main()

x: 2.0 a: 3.0  
y: 4.0

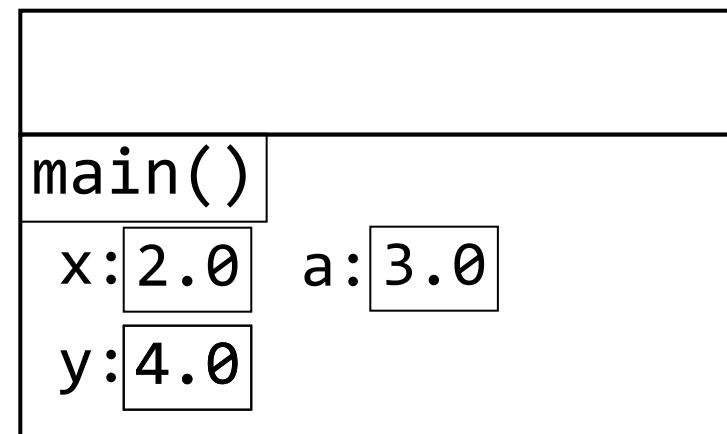
```
void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}
```

The square of 2 is 4



```

void printSquare(double a)
{
    printf("The square of %g is %g", a, square(a));
}

double square(double a)
{
    double result = a * a;
    return result;
}

int main(void)
{
    double x = 2.0;
    double y = 4.0;
    double a = average(x, y);
    printSquare(x);

    return 0;
}

```

The square of 2 is 4

