

Tutorial 5

Group 01A 02A

2 October 2024

Problem 13.1

```
long square(long x)
{
    return x*x;
}
```

```
double hypotenuse_of(long base, long height)
{
    return sqrt(square(base) + square(height));
}
```

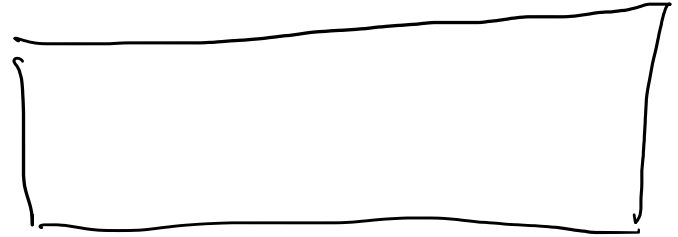
```
int main()
{
    hypotenuse_of(3, 4);
}
```

```
long square(long x)
{
    return x*x;
}
```

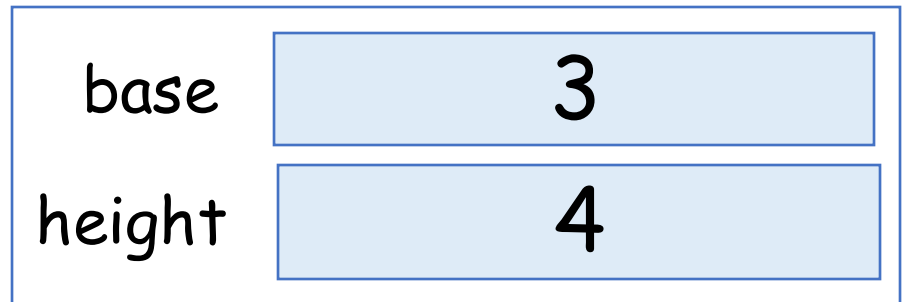
```
double hypotenuse_of(long base, long height)
{
    return sqrt(square(base) + square(height));
}
```

```
int main()
{
    hypotenuse_of(3, 4);
}
```

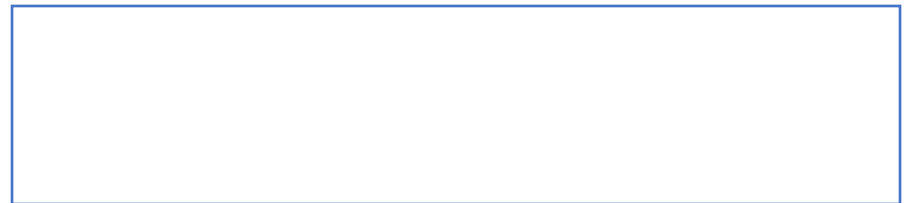
main



hypotenuse_of



main



```
long square(long x)
```

```
{  
  return x*x;  
}
```

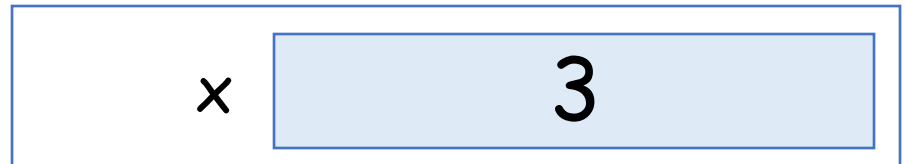
```
double hypotenuse_of(long base, long height)
```

```
{  
  return sqrt(square(base) + square(height));  
}
```

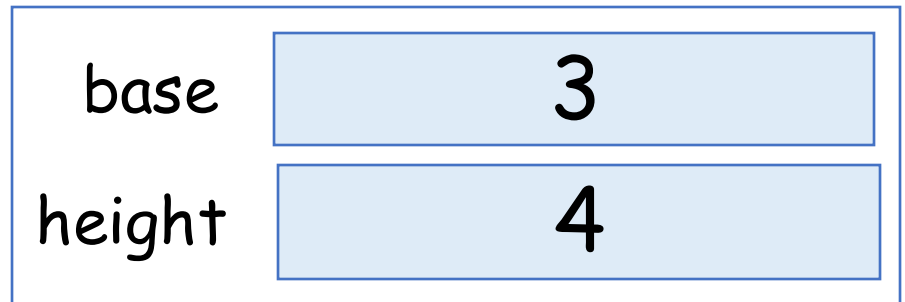
```
int main()
```

```
{  
  hypotenuse_of(3, 4);  
}
```

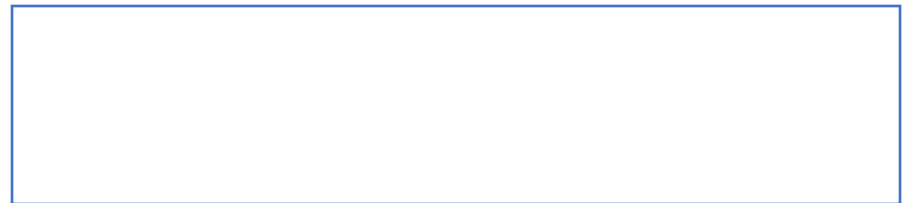
square



hypotenuse_of



main



```
long square(long x)
{
    return x*x;
}
```

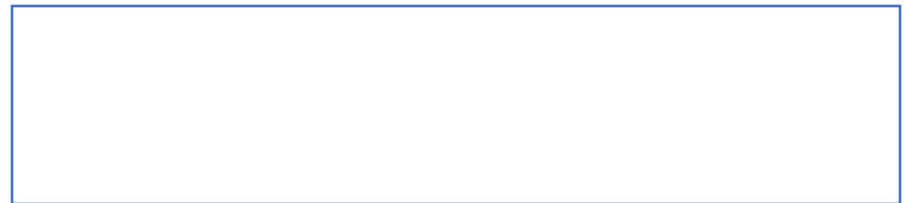
```
double hypotenuse_of(long base, long height)
{
    return sqrt(square(base) + square(height));
}
```

```
int main()
{
    hypotenuse_of(3, 4);
}
```

hypotenuse_of

base	3
height	4

main



```
long square(long x)
```

```
{  
  return x*x;  
}
```

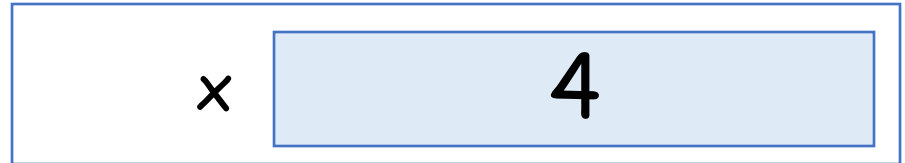
```
double hypotenuse_of(long base, long height)
```

```
{  
  return sqrt(square(base) + square(height));  
}
```

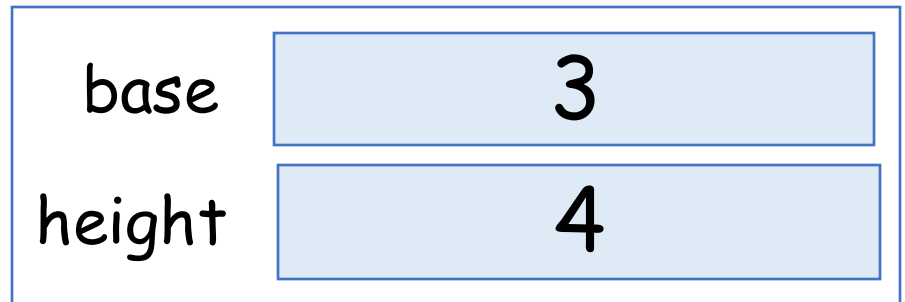
```
int main()
```

```
{  
  hypotenuse_of(3, 4);  
}
```

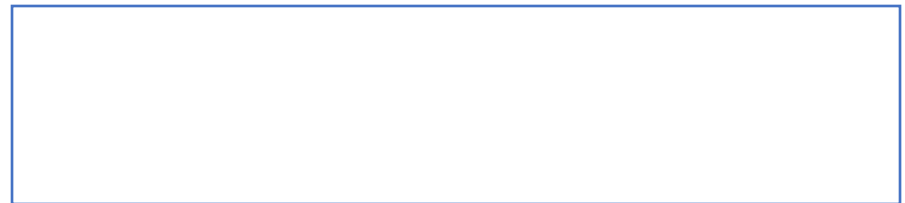
square



hypotenuse_of



main



```
long square(long x)
{
    return x*x;
}
```

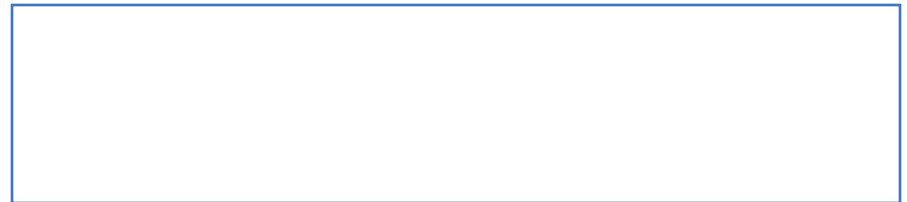
```
double hypotenuse_of(long base, long height)
{
    return sqrt(square(base) + square(height));
}
```

```
int main()
{
    hypotenuse_of(3, 4);
}
```

hypotenuse_of

base	3
height	4

main




```
long square(long x)
{
    return x*x;
}
```

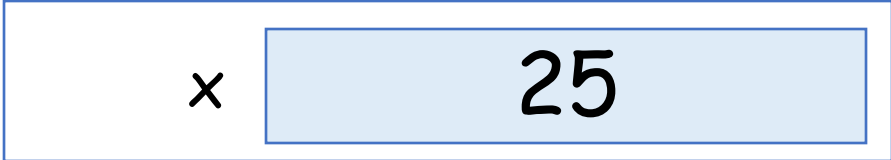
```
double hypotenuse_of(long base, long height)
{
    return sqrt(square(base) + square(height));
}
```

```
int main()
{
    hypotenuse_of(3, 4);
}
```

GNU C Lib

Sqrt(x) { ... }

sqrt



hypotenuse_of



main



```
long square(long x)
{
    return x*x;
}
```

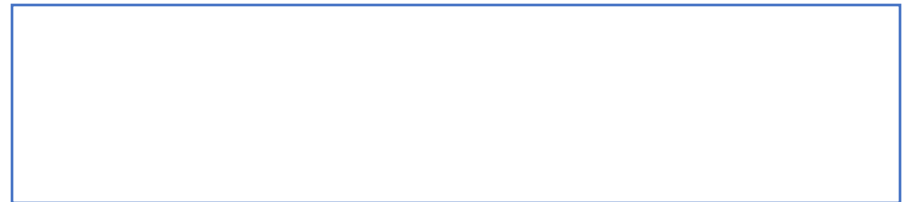
```
double hypotenuse_of(long base, long height)
{
    return sqrt(square(base) + square(height));
}
```

```
int main()
{
    hypotenuse_of(3, 4);
}
```

hypotenuse_of

base	3
height	4

main



```
long square(long x)
```

```
{  
    return x*x;  
}
```

```
double hypotenuse_of(long base, long height)
```

```
{  
    return sqrt(square(base) + square(height));  
}
```

```
int main()
```

```
{  
    hypotenuse_of(3, 4);  
}
```

main



```
long square(long x)
```

```
{  
    return x*x;  
}
```

```
double hypotenuse_of(long base, long height)
```

```
{  
    return sqrt(square(base) + square(height));  
}
```

```
int main()
```

```
{  
    hypotenuse_of(3, 4);  
}
```

Problem 13.2

```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
  
int main() {  
    factorial(3);  
}
```

```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
int main(){  
    factorial(3);  
}
```

factorial

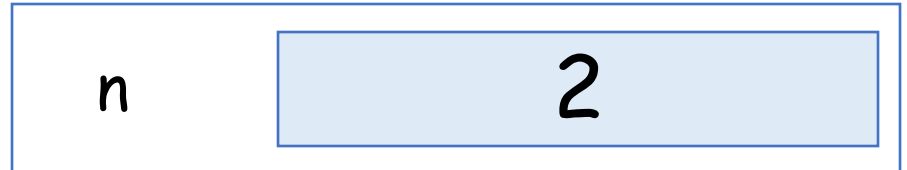
n

3

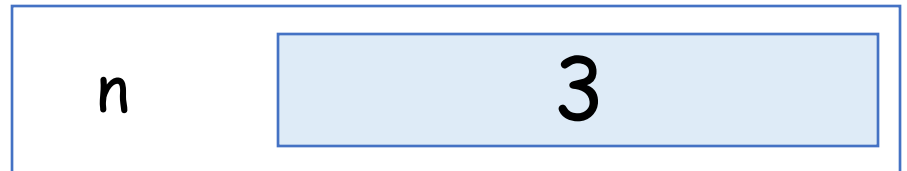
main

```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
  
int main(){  
    factorial(3);  
}
```

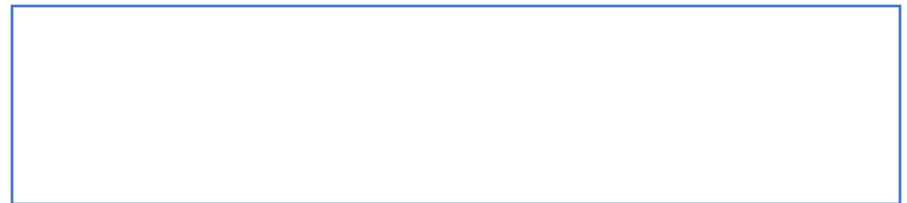
factorial



factorial



main




```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
  
int main(){  
    factorial(3);  
}
```

factorial

n

1

factorial

n

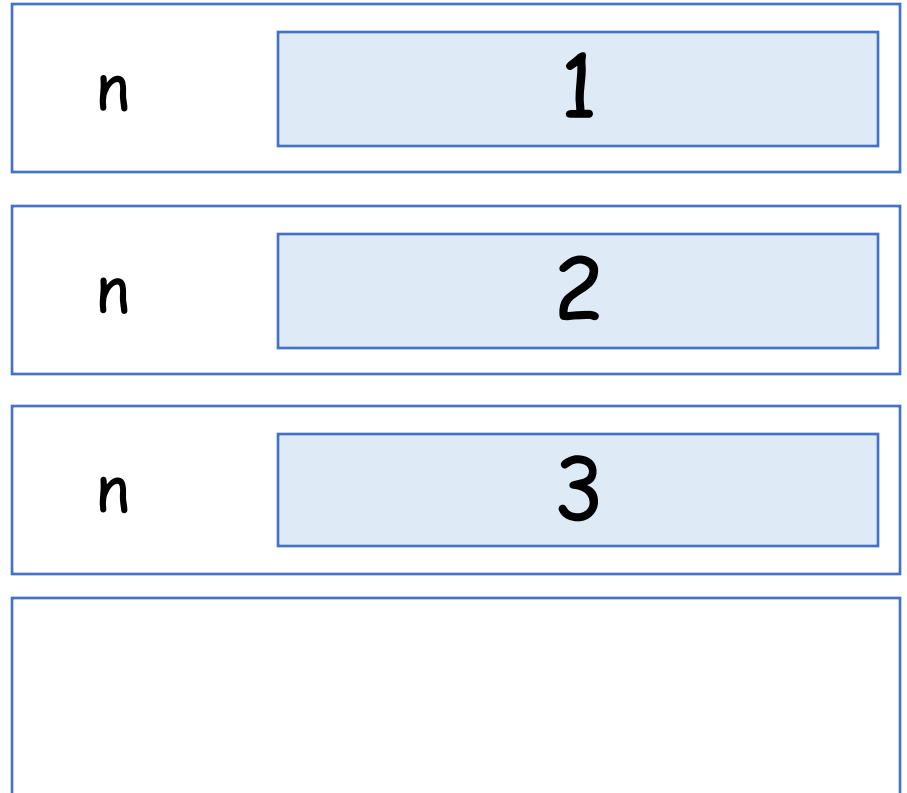
2

factorial

n

3

main



```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
int main(){  
    factorial(3);  
}
```

factorial

n

0

factorial

n

1

factorial

n

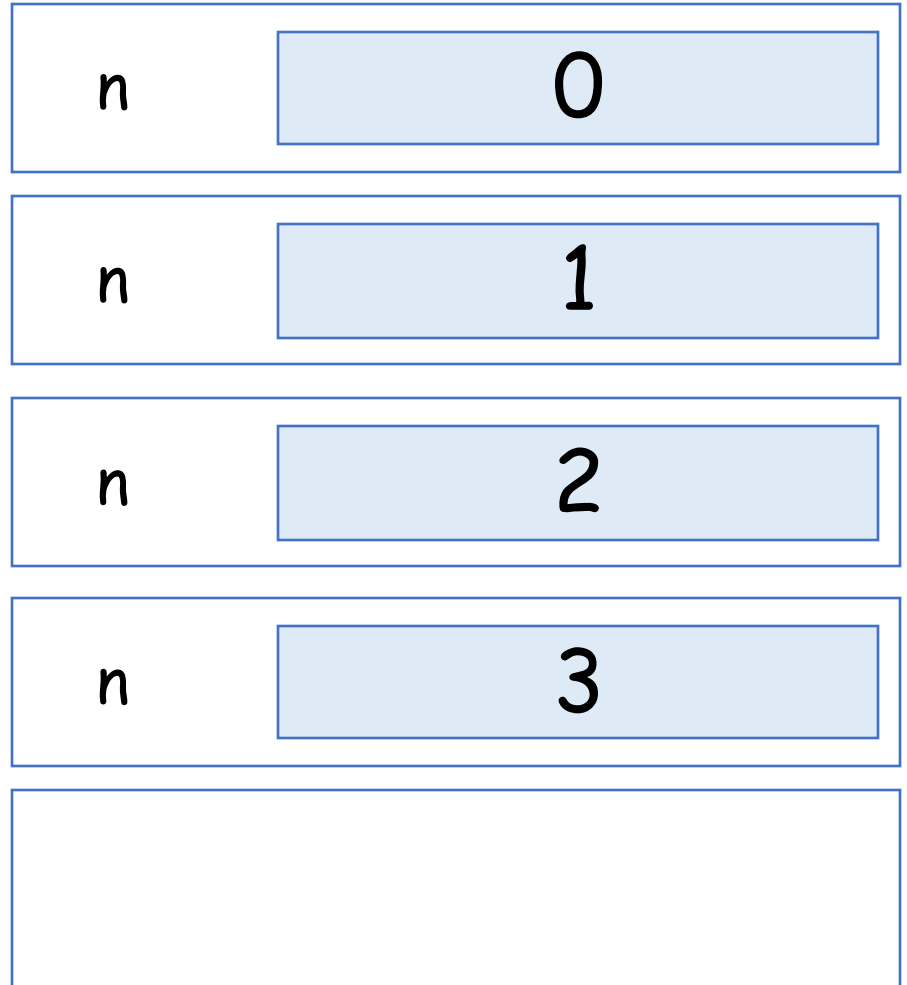
2

factorial

n

3

main



```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
  
int main(){  
    factorial(3);  
}
```

factorial

n

1

factorial

n

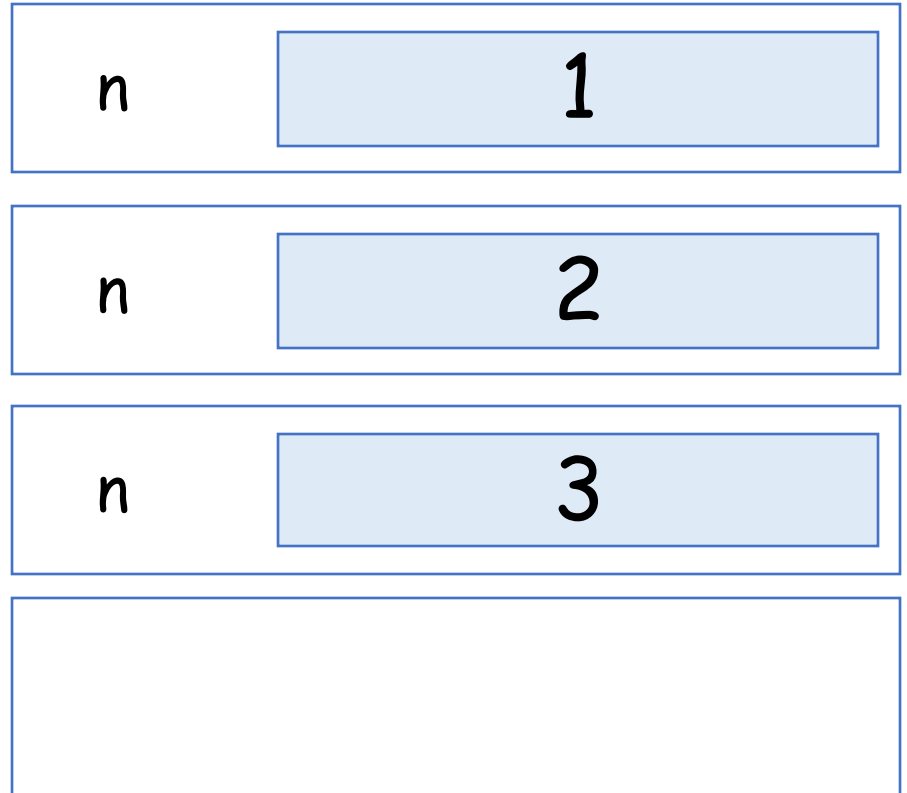
2

factorial

n

3

main

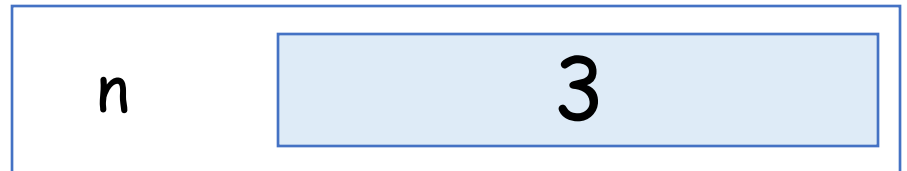


```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
  
int main(){  
    factorial(3);  
}
```

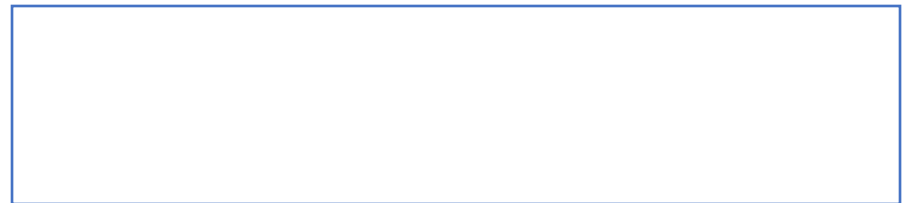
factorial



factorial



main



```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
int main(){  
    factorial(3);  
}
```

factorial

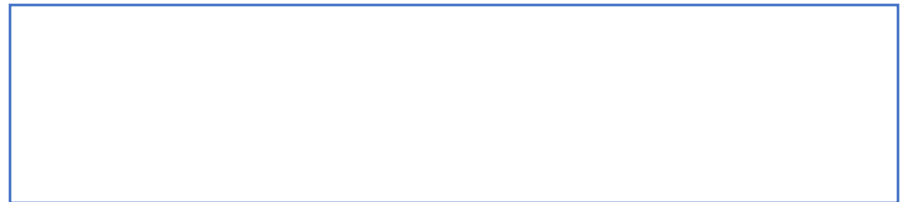
n

3

main

```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
int main(){  
    factorial(3);  
}
```

main



```
long factorial(long n) {  
    if (n == 0) {  
        return 1;  
    }  
    return factorial(n-1) * n;  
}  
  
int main(){  
    factorial(3);  
}
```

Problem 14.1


```
void doThat(long a[], long b[]) {
```

```
    a[0] = 100;
```

```
    b[1] = 200;
```

```
    // Line A
```

```
}
```

```
void doThis(long a[]) {
```

```
    long *b = a;
```

```
    doThat(a, b);
```

```
}
```

```
int main() {
```

```
    long a[3] = {0, 0, 0};
```

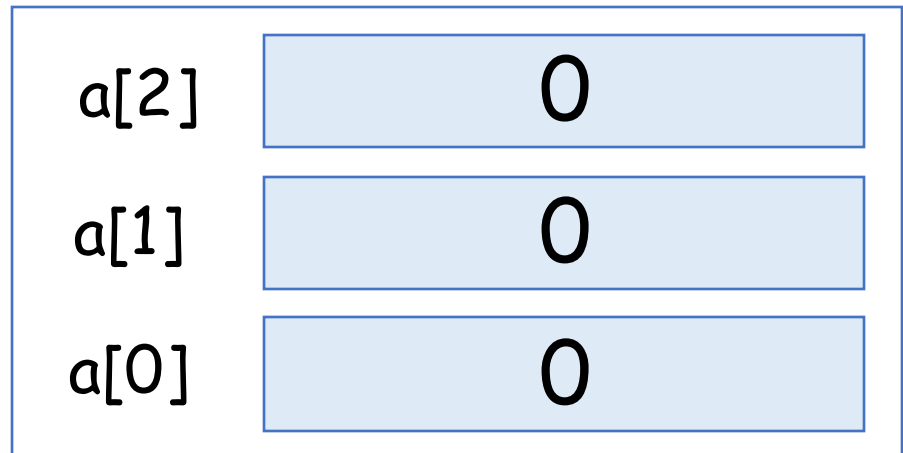
```
    doThis(a);
```

```
    // Line B
```

```
}
```

```
void doThat(long a[], long b[]) {  
    a[0] = 100;  
    b[1] = 200;  
    // Line A  
}  
  
void doThis(long a[]) {  
    long *b = a;  
    doThat(a, b);  
}  
  
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

main

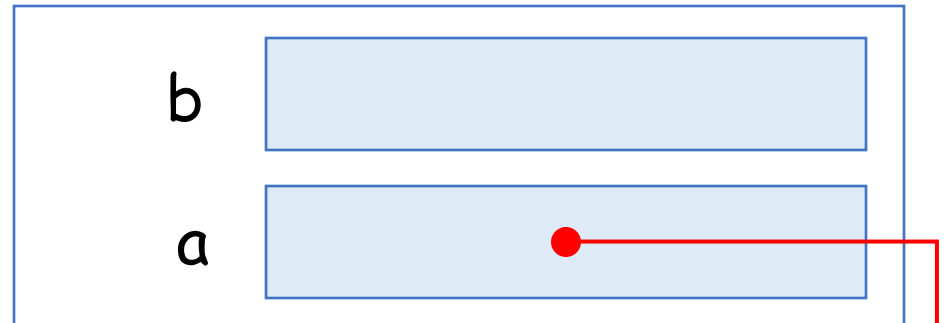


```
void doThat(long a[], long b[]) {  
    a[0] = 100;  
    b[1] = 200;  
    // Line A  
}
```

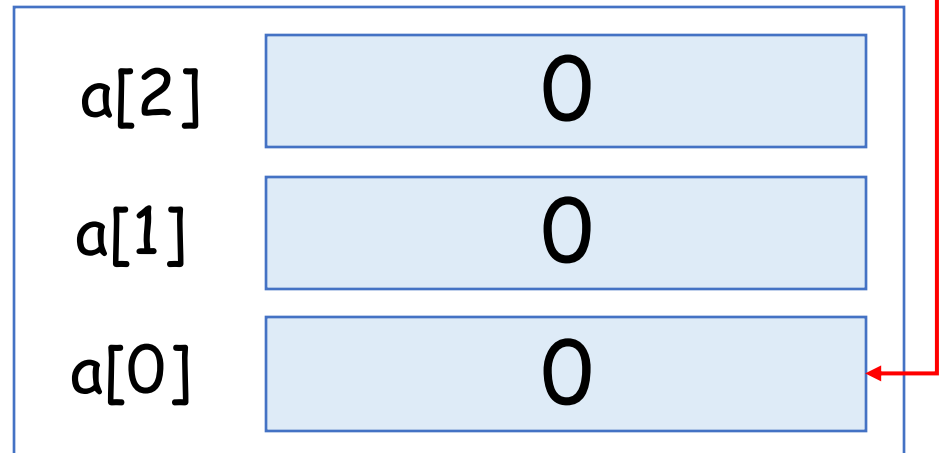
```
void doThis(long a[]) {  
    long *b = a;  
    doThat(a, b);  
}
```

```
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

doThis



main

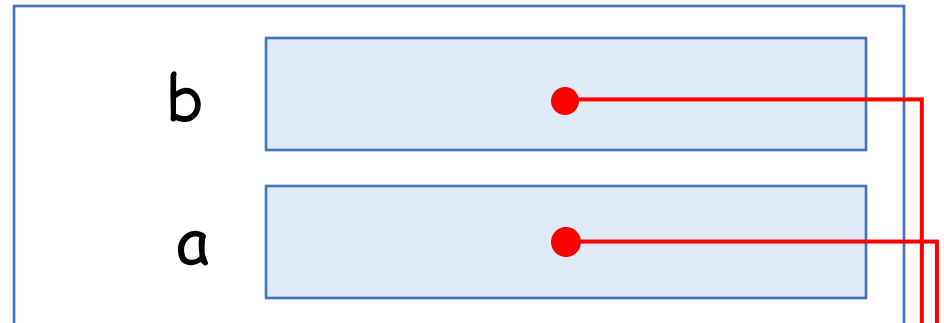


```
void doThat(long a[], long b[]) {  
    a[0] = 100;  
    b[1] = 200;  
    // Line A  
}
```

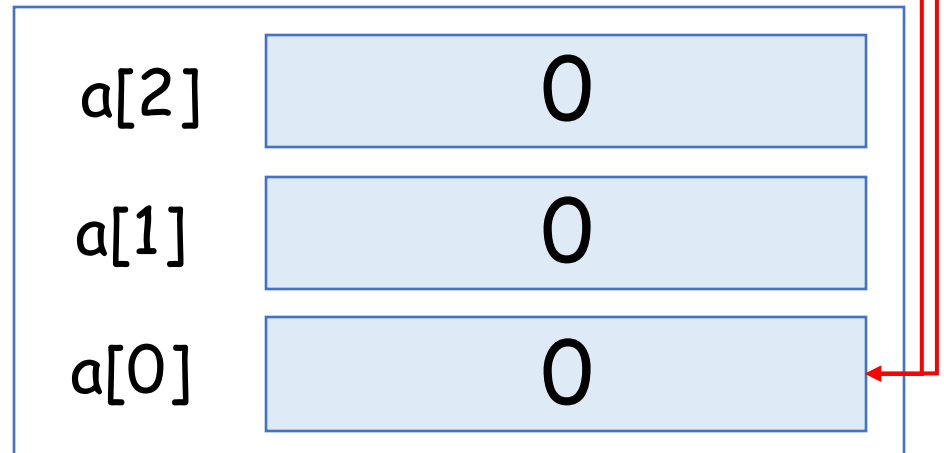
```
void doThis(long a[]) {  
    long *b = a;  
    doThat(a, b);  
}
```

```
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

doThis



main



```
void doThat(long a[], long b[]) {
```

```
    a[0] = 100;
```

```
    b[1] = 200;
```

```
    // Line A
```

```
}
```

```
void doThis(long a[]) {
```

```
    long *b = a;
```

```
    doThat(a, b);
```

```
}
```

```
int main() {
```

```
    long a[3] = {0, 0, 0};
```

```
    doThis(a);
```

```
    // Line B
```

```
}
```

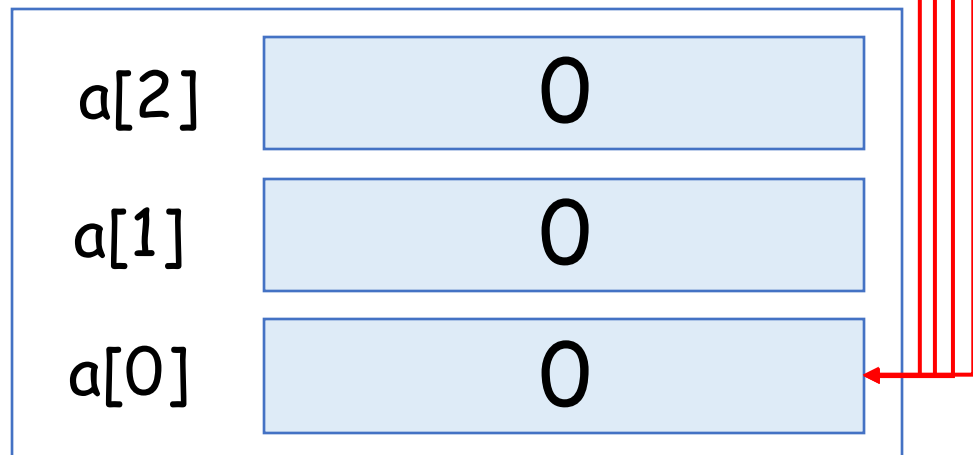
doThat



doThis



main



```
void doThat(long a[], long b[]) {
```

```
    a[0] = 100;
```

```
    b[1] = 200;
```

```
    // Line A
```

```
}
```

```
void doThis(long a[]) {
```

```
    long *b = a;
```

```
    doThat(a, b);
```

```
}
```

```
int main() {
```

```
    long a[3] = {0, 0, 0};
```

```
    doThis(a);
```

```
    // Line B
```

```
}
```

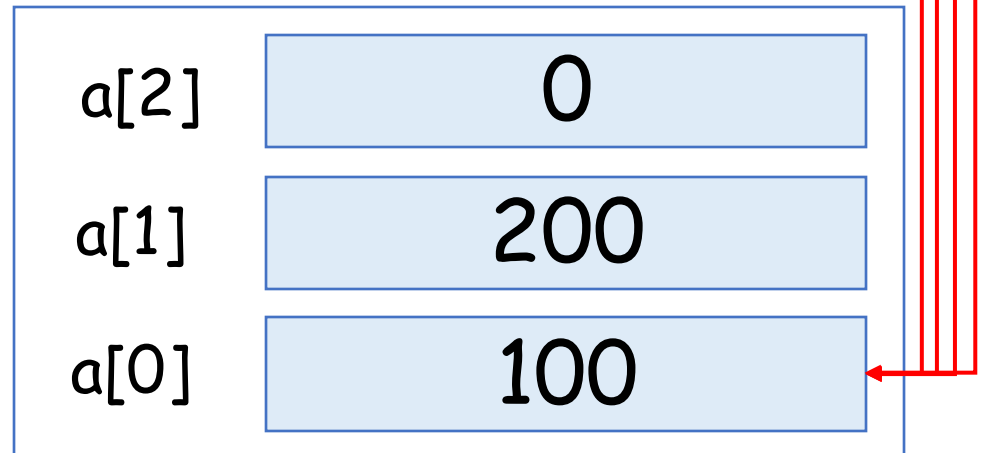
doThat



doThis



main



```
void doThat(long a[], long b[]) {  
    a[0] = 100;  
    b[1] = 200;  
    // Line A  
}  
  
void doThis(long a[]) {  
    long *b = a;  
    doThat(a, b);  
}  
  
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

main

a[2]	0
a[1]	200
a[0]	100

Problem 14.2


```
void doThat(long list[]) {
```

```
    list[1] = 200;
```

```
    // Line A
```

```
}
```

```
void doThis(const long a[]) {
```

```
    long b[2] = {10, 10};
```

```
    a = b;
```

```
    doThat(a);
```

```
}
```

```
int main() {
```

```
    long a[3] = {0, 0, 0};
```

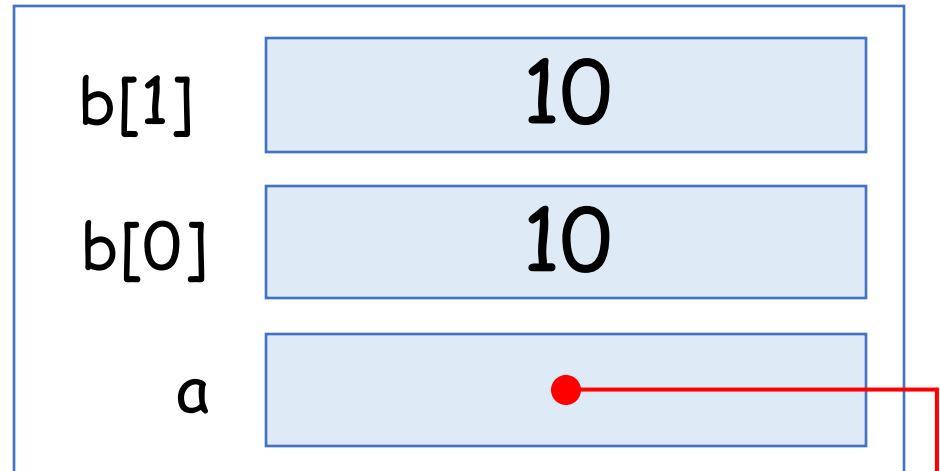
```
    doThis(a);
```

```
    // Line B
```

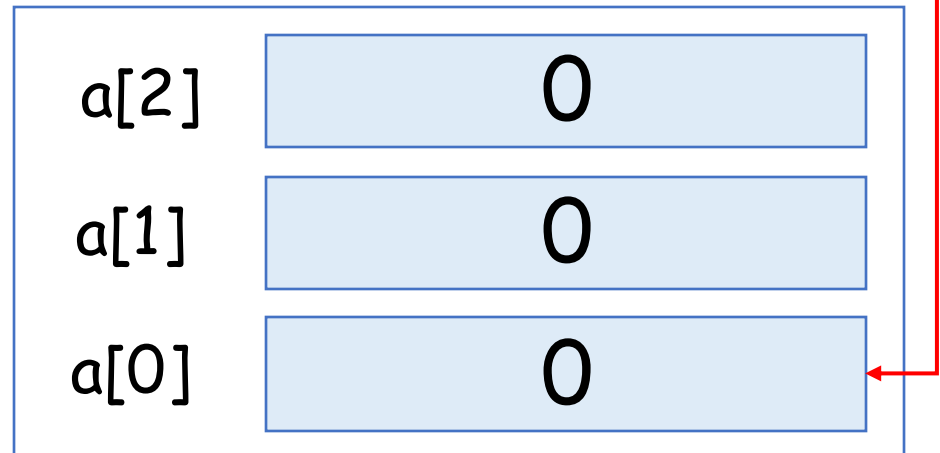
```
}
```

```
void doThat(long list[]) {  
    list[1] = 200;  
    // Line A  
}  
  
void doThis(const long a[]) {  
    long b[2] = {10, 10};  
    a = b;  
    doThat(a);  
}  
  
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

doThis

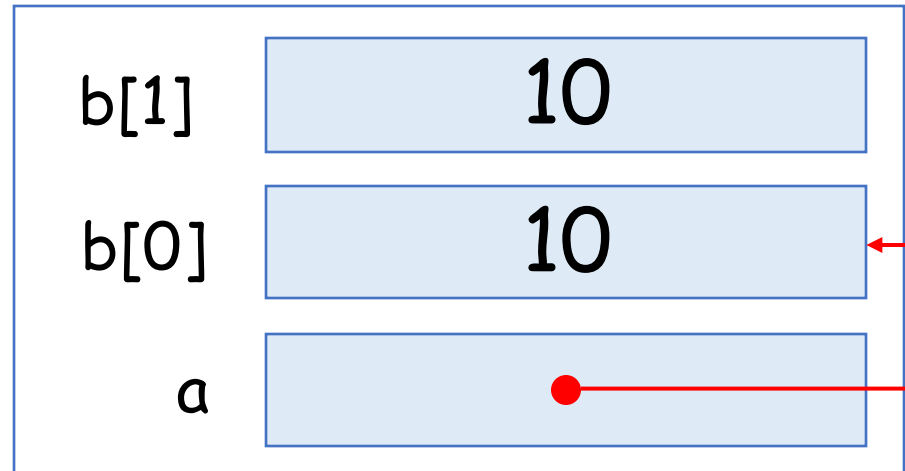


main

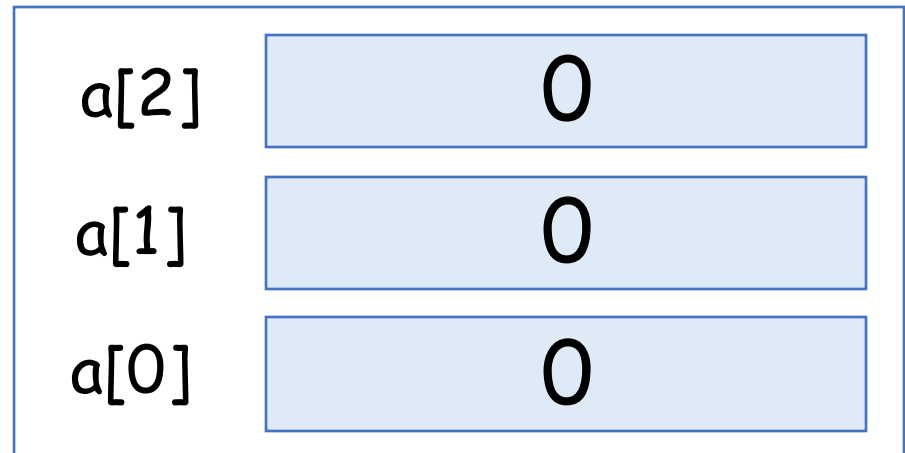


```
void doThat(long list[]) {  
    list[1] = 200;  
    // Line A  
}  
  
void doThis(const long a[]) {  
    long b[2] = {10, 10};  
    a = b;  
    doThat(a);  
}  
  
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

doThis

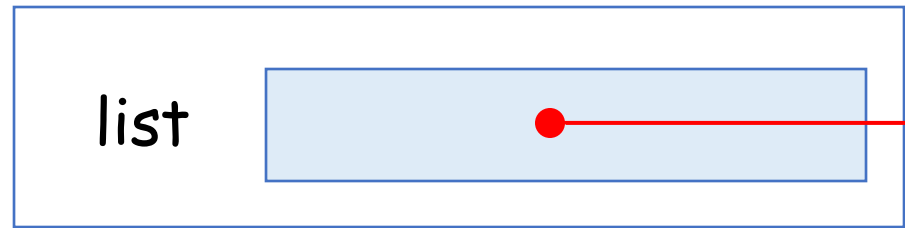


main

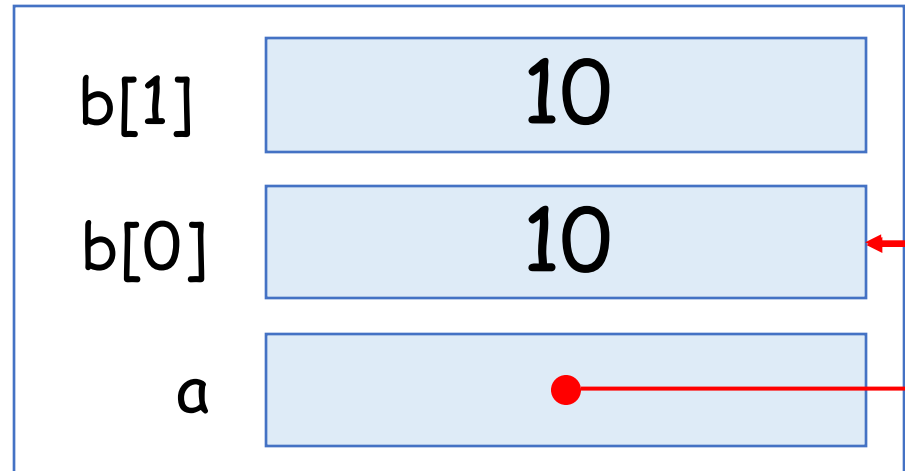


```
void doThat(long list[]) {  
    list[1] = 200;  
    // Line A  
}  
  
void doThis(const long a[]) {  
    long b[2] = {10, 10};  
    a = b;  
    doThat(a);  
}  
  
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

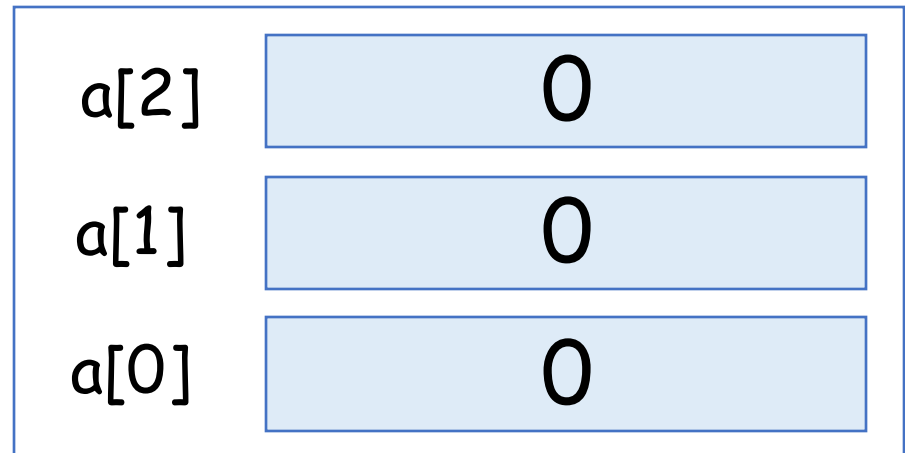
doThat



doThis

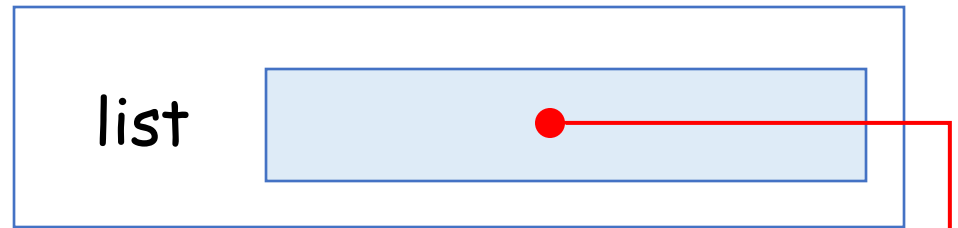


main

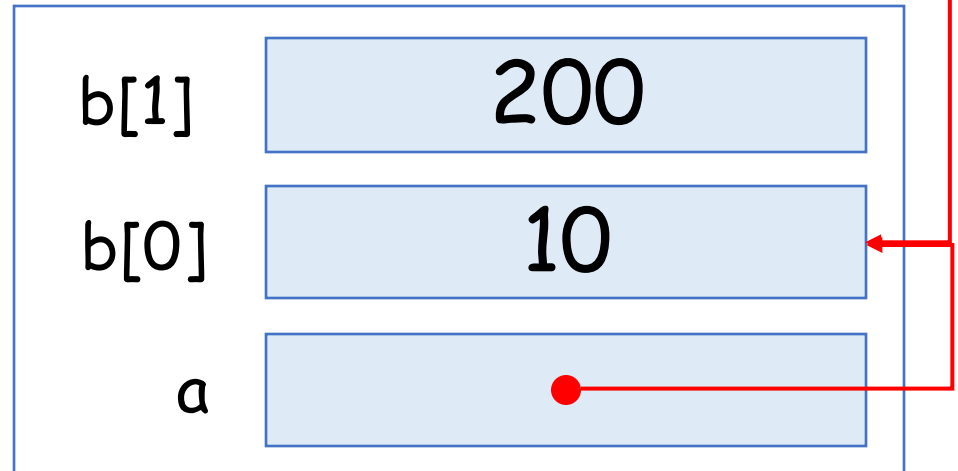


```
void doThat(long list[]) {  
    list[1] = 200;  
    // Line A  
}  
  
void doThis(const long a[]) {  
    long b[2] = {10, 10};  
    a = b;  
    doThat(a);  
}  
  
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

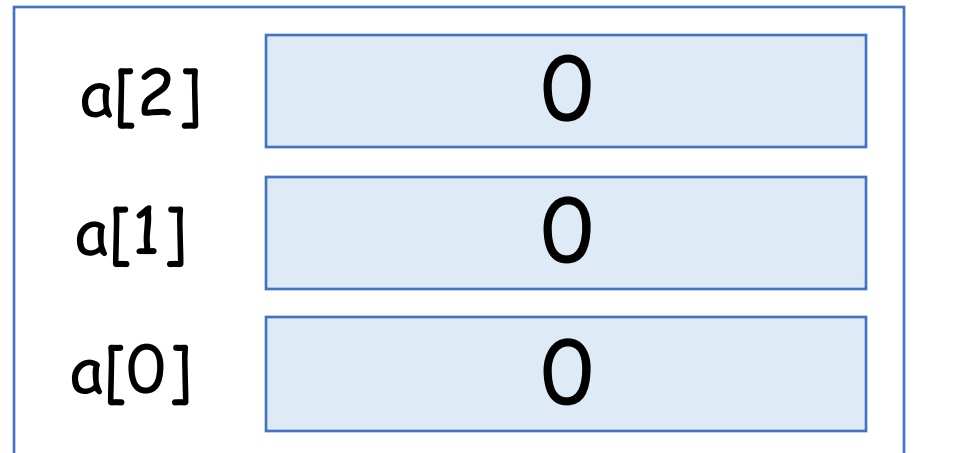
doThat



doThis



main



```
void doThat(long list[]) {  
    list[1] = 200;  
    // Line A  
}  
  
void doThis(const long a[]) {  
    long b[2] = {10, 10};  
    a = b;  
    doThat(a);  
}  
  
int main() {  
    long a[3] = {0, 0, 0};  
    doThis(a);  
    // Line B  
}
```

main

