APS105
Winter 2012

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

Lecture 9
February 3, 2012

Today

- More loop examples
- `break` and `continue`
- Arrays

- Assignment 1 due Saturday
- CodeLab 3 due Sunday

```
   ***
   ***
   ***
   ***
```

height: 4

width: 3

height rows

width "*" per row for all rows

```
for (int y = 0; y < height; y++)
{
  for (int x = 0; x < width; x++)
  {
      printf("*");
  }

  printf("\n");
}
```

```
*
**
***
****
```

height: 4

height rows

y "*" in row y

```
for (int y = 0; y < height; y++)
{
  for (int x = 0; x < y; x++)
  {
      printf("*");
  }

  printf("\n");
}
```

```
*
**
***
```

```
*
**
***
****
```

height: 4

height rows

y "*" in row y (1-based index)

```
for (int y = 1; y <= height; y++)
{
  for (int x = 0; x < y; x++)
  {
      printf("*");
  }

  printf("\n");
}
```

```
****
***
**
*
```

```
height: 4
```

height rows

height - y "*" in row y

(0-based index)

```
for (int y = 0; y < height; y++)
{
  for (int x = 0; x < height - y; x++)
  {
      printf("*");
  }

  printf("\n");
}
```

```
   *
  **
 ***
****
```

`height: 4`

`height` rows

`height - y " "` and

`y "*"` in row `y` (1-based index)

	Row	# of " "	# of "*"
*	1	3	1
**	2	2	2
***	3	1	3
****	4	0	4

```
   *
  **
 ***
****
```

height: 4

height rows

height - y " " and y "*" in row y

```
for (int y = 1; y <= height; y++)
{

    for (int x = 0; x < height - y; x++)
    {
        printf(" ");
    }

    for (int x = 0; x < y; x++)
    {
        printf("*");
    }

    printf("\n");
}
```

```
   *
  **
 ***
****
```

height: 4

height rows

height - y " " and y "*" in row y

```
for (int y = 1; y <= height; y++)
{
    int x;
    for (x = 0; x < height - y; x++)
    {
        printf(" ");
    }

    for ( ; x < height; x++)
    {
        printf("*");
    }

    printf("\n");
}
```

break

- Sometimes you want to end a loop early

```
bool keepGoing = true;
while (keepGoing)
{
  int num;
  printf("Enter a number (0 to quit): ");
  scanf("%d", &num);

  if (num == 0)
  {
    keepGoing = false;
  }
  else
  {
    int square = num * num;
    printf("%d squared is %d \n", num, square);
  }
}
```

```
while (true)
{
   int num;
   printf("Enter a number (0 to quit): ");
   scanf("%d", &num);

   if (num == 0)
   {
      break;
   }


   int square = num * num;
   printf("%d squared is %d \n", num, square);

}
```

break

- Sometimes you want to end a loop early
- This occurs less frequently than you think

```
while (true)
{
   if (n >= MAX)
   {
      break;
   }
   // do something
}
```

```
while (n >= MAX)
{
   // do something
}
```

break

- Sometimes you want to end a loop early
- This occurs less frequently than you think

- break can make code much harder to read
- Unless indicated otherwise, you are not permitted to use break on assignments and tests

continue

- Skips the rest of this iteration
- Similar issues as break
- Unless indicated otherwise, you are not permitted to use continue on assignments and tests

```
for (int i = 0; i < 10; i++)
{
    if (i % 2 == 0)
    {
        continue;
    }
    printf("%d\n", i);
}
```

```
1
3
5
7
9
```

Arrays

23

47

39

39 47 23

```
int first = 0;
int second = 0;
int third = 0;

printf("Enter a number: ");
scanf("%d", &first);

printf("Enter a number: ");
scanf("%d", &second);

printf("Enter a number: ");
scanf("%d", &third);

printf("%d %d %d\n", third, second, first);
```

23
47
39

```
39 47 23
```

```
#define N 3
...

int first = 0;
int second = 0;
int third = 0;


for (int i = 0; i < N; i++)
{
    printf("Enter a number: ");
    scanf("%d", ??? );
}

for (int i = 0; i < N; i++)
{
    printf("%d ", ??? );
}
```

23

47

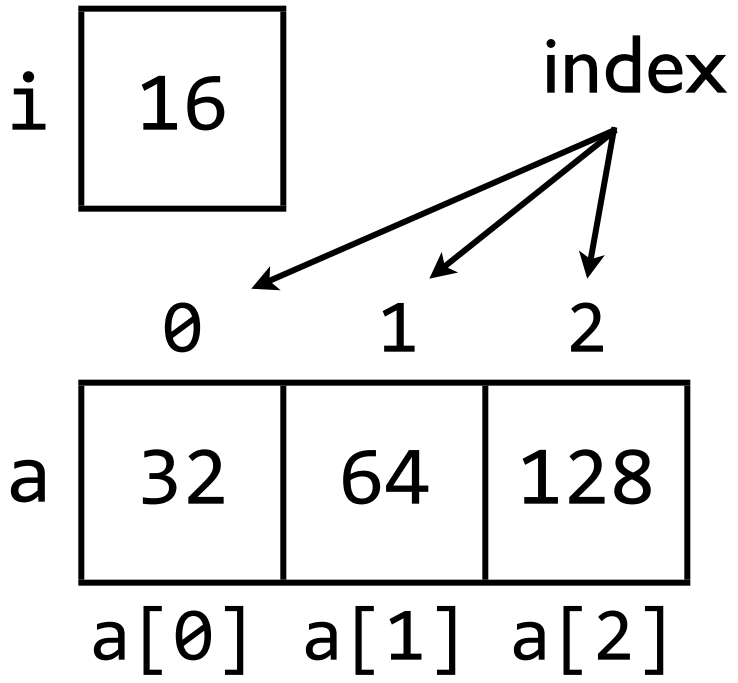
39

```
39 47 23
```

Array

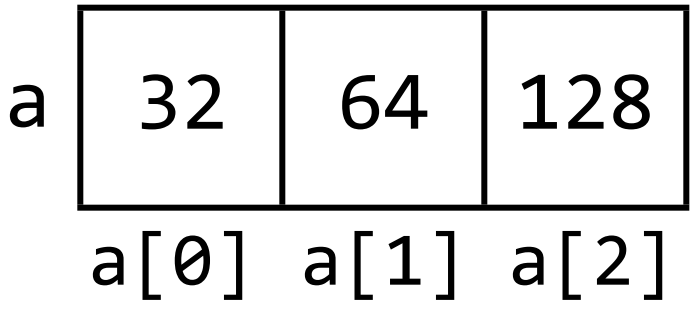
- A connected group of variables

```
int i;
i = 16;
```




```
int a[3];
a[0] = 32;
a[1] = 64;
a[2] = 128;
```

```
int a[3];
a[0] = 32;
a[1] = 64;
a[2] = 128;
```



What's the type of a? array of `int`

What's the type of `a[0]`? `int`

What's the type of `a[1]`? `int` } lvalues

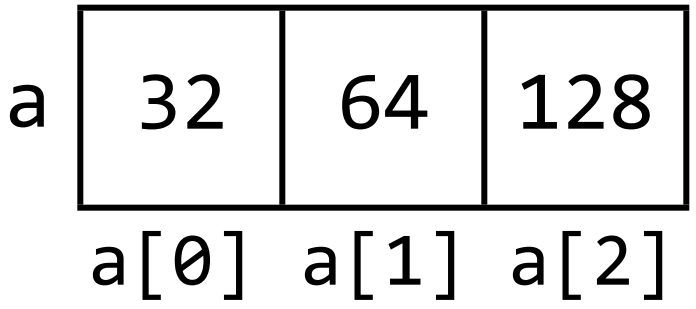
What's the type of `a[2]`? `int`

```
int x = 9;
a[0] = x + 2;
int sum = a[0] + a[1];
a[0]++;
a = 14;
```

Error

```
if (a[0] > MAX)
{
    ...
}
```

```
int a[3];
a[0] = 32;
a[1] = 64;
a[2] = 128;
```



What's the type of a? array of `int`

What's the type of `a[0]`? `int`

What's the type of `a[1]`? `int`

What's the type of `a[2]`? `int`

} lvalues

To get an `int`

`i`

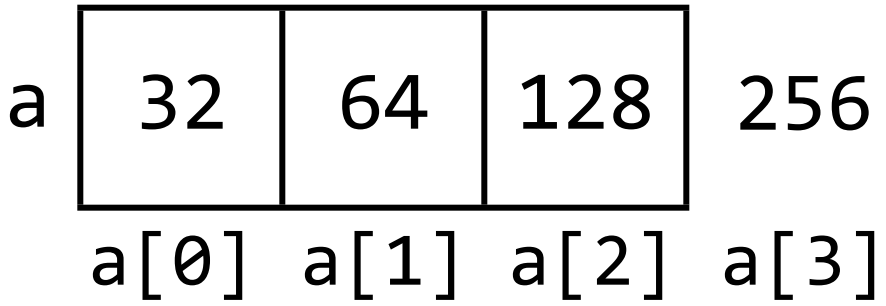
`a[0]`

```
int[3] a;
```

Error

```
int i;
int a[2];
int x, y, z[5];
```

```
int a[3];
a[0] = 32;
a[1] = 64;
a[2] = 128;
a[3] = 256;
```



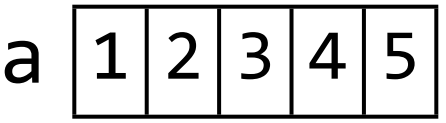
Initializing Arrays

```
int i;
i = 5;
```

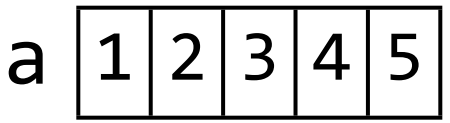
```
int i = 5;
```

```
int a[5];
a[0] = 1;
a[1] = 2;
a[2] = 3;
a[3] = 4;
a[4] = 5;
```

```
int a[5] = {1, 2, 3, 4, 5};
```




```
int a[]  = {1, 2, 3, 4, 5};
```



```
int a[5] = {1, 2};
```
a

1	2	0	0	0

```
int a[5] = {1, 2, 0, 0, 0};
```

```
int a[]  = {1, 2};
```
a

1	2

```
int a[5] = {};
```
Error

```
int a[5] = {0};
```
a

0	0	0	0	0

```
int a[5] = {1, 2, 3, 4, 5, 6, 7};
```
Error

```
int a[5] = {1, 2, 3, 4, 5};

a = {6, 7, 8, 9, 10};
```

Error

Length of Arrays

```
int a[int-expression];
```

```
int a[10];
```

```
int a[2 * 5];
```

```
int a[6.5];
```
Error

```
int a[6.0];
```
Error

```
int a[N];
```
It depends...

Length of Arrays

- Three types of arrays:
  - Static
    - Length must be a known constant
  - Variable (new in C99)
    - Length can use variables
  - Dynamic
    - Later

```
#define N 8
...
int i;
scanf("%d", &i);
```

```
int a[10];
```
Static

```
int b[N];
```
Static

```
int c[i];
```
Variable

```
int d[2 * N];
```
Static

```
int e[2 * i];
```
Variable

Iterating Arrays

```
#define N 10
...
int a[N];

for (int i = 0; i < N; i++)
{
   a[i] = i;
}

for (int i = 0; i < N; i++)
{
   printf("%d\n", a[i]);
}
```

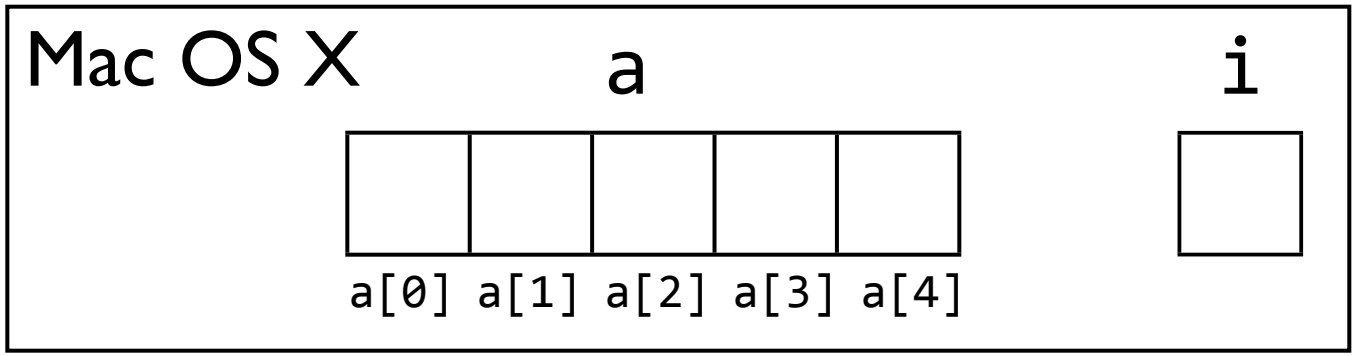
Subscripts

- Any `int` expression
- Must be valid (in range), though

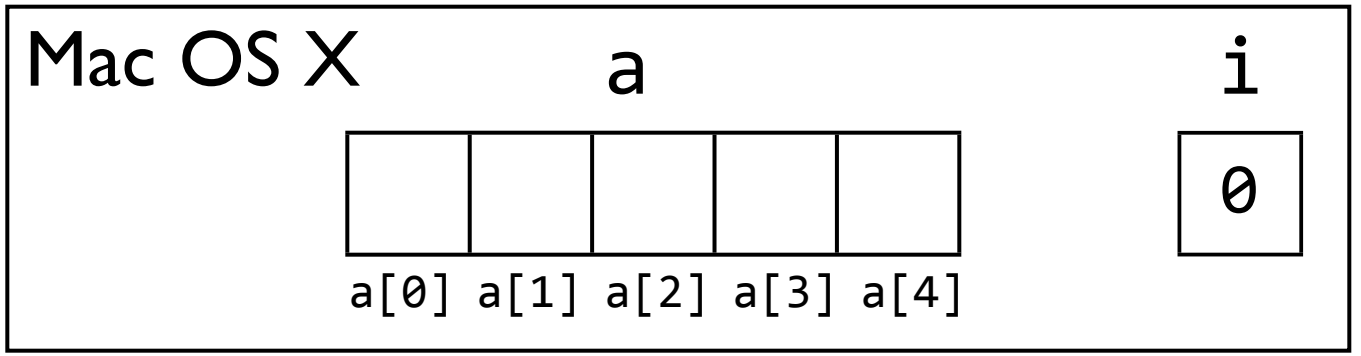
```
int result;
result = a[0];
result = a[i];
result = a[i + 2];
result = a[2 * i + 4 / j - x + 22 % t];

result = a[2.0];
```

Bad Style

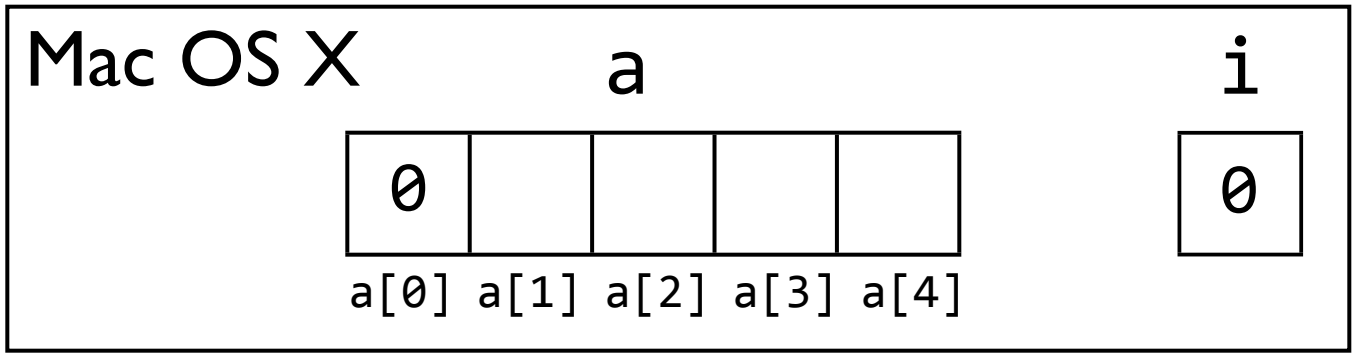
Error



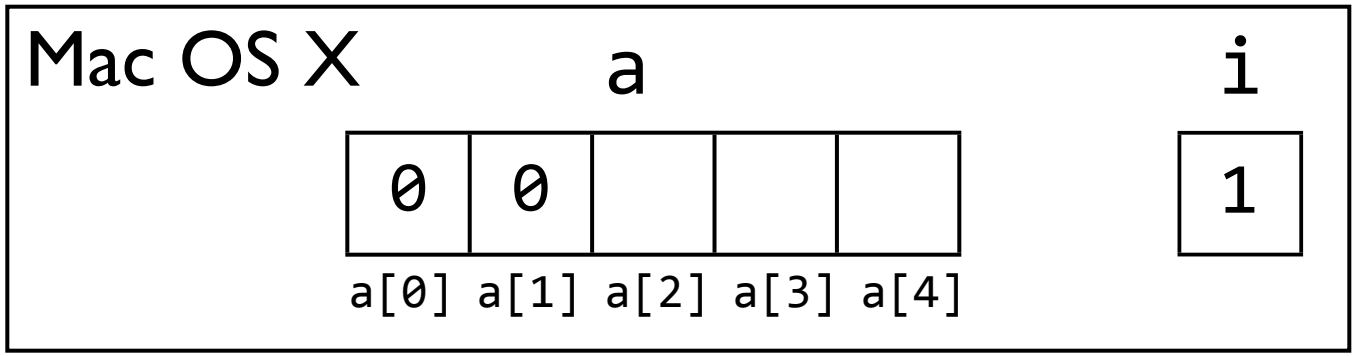

```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



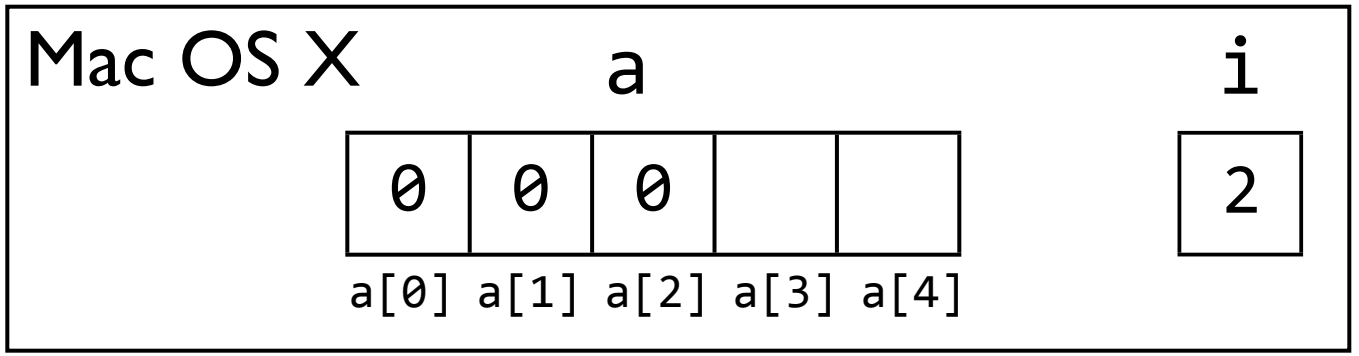

```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



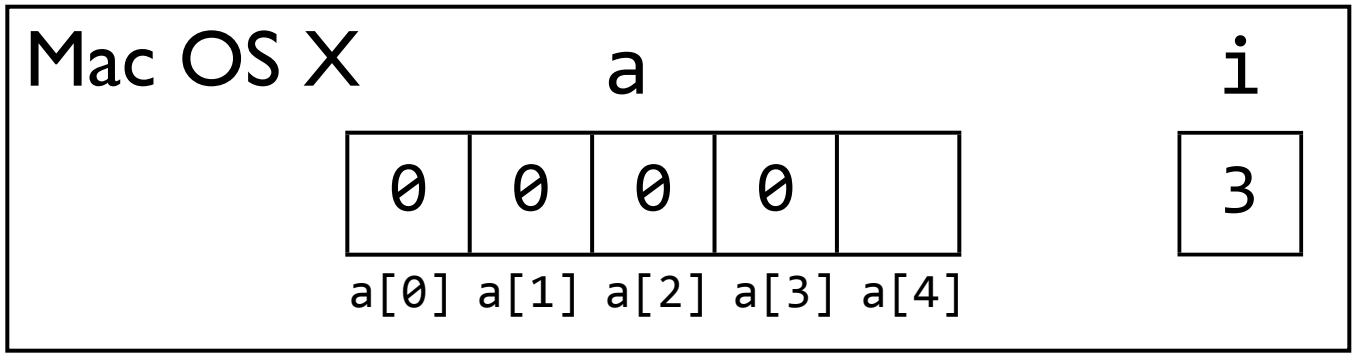

```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



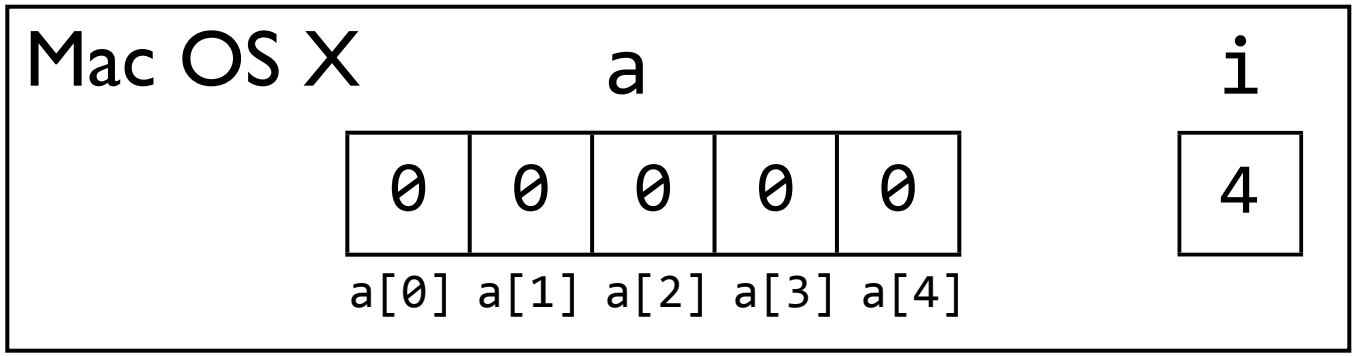

```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



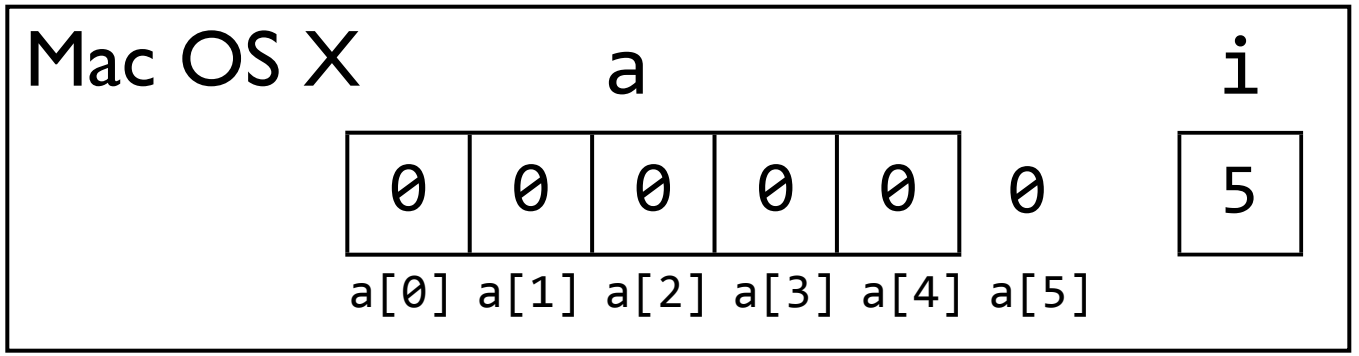

```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



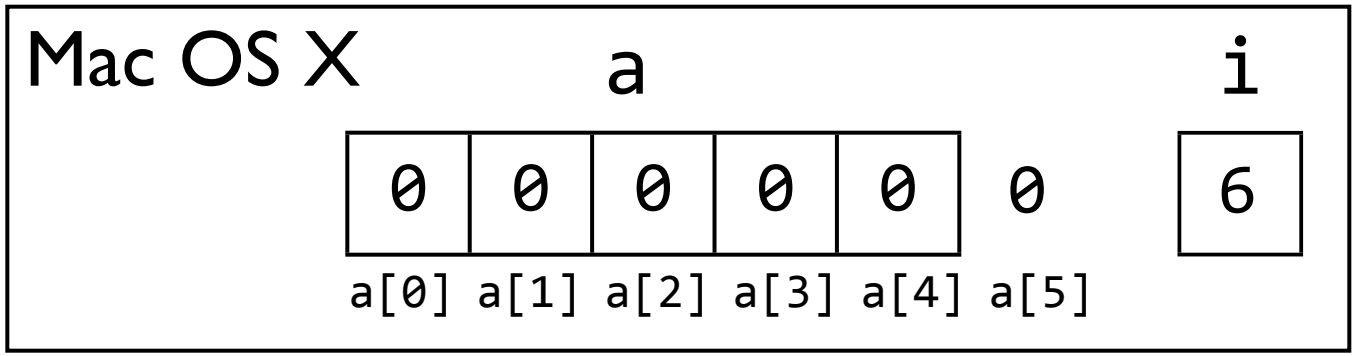

```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```




```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```

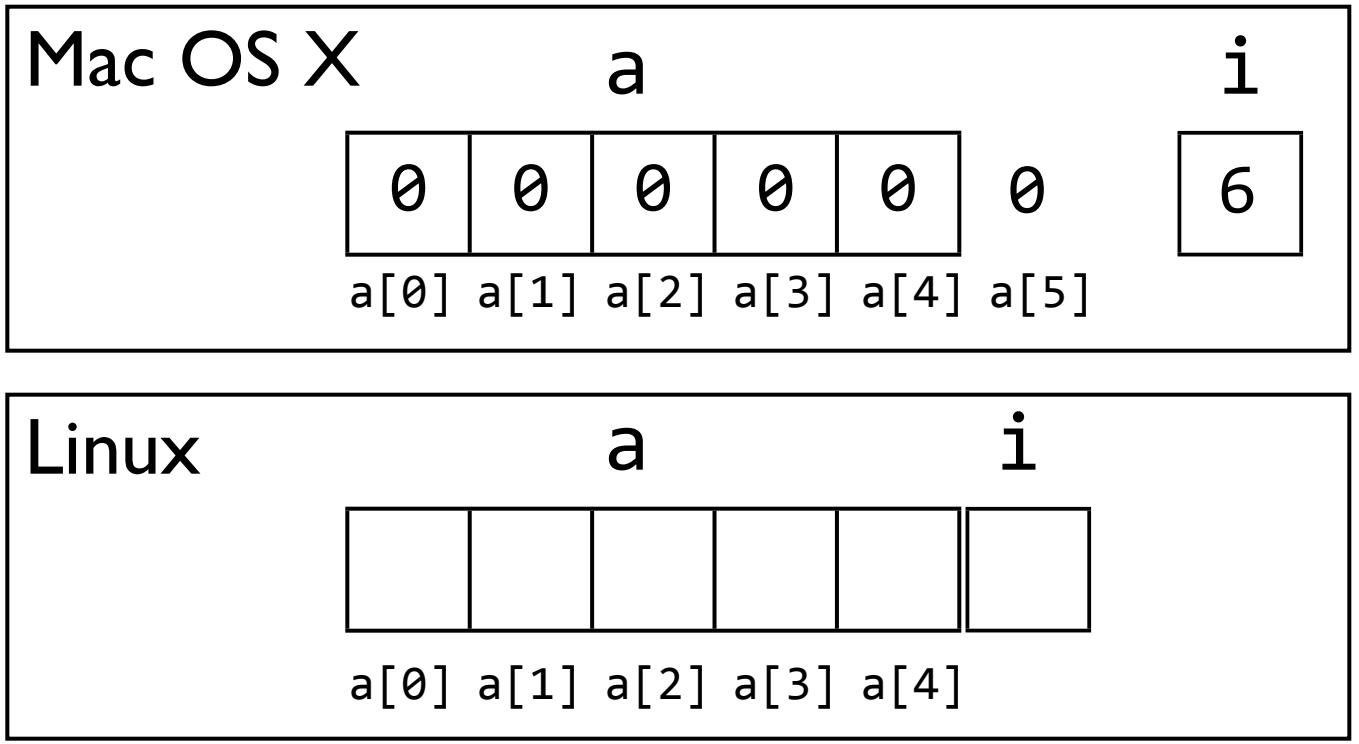



```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```

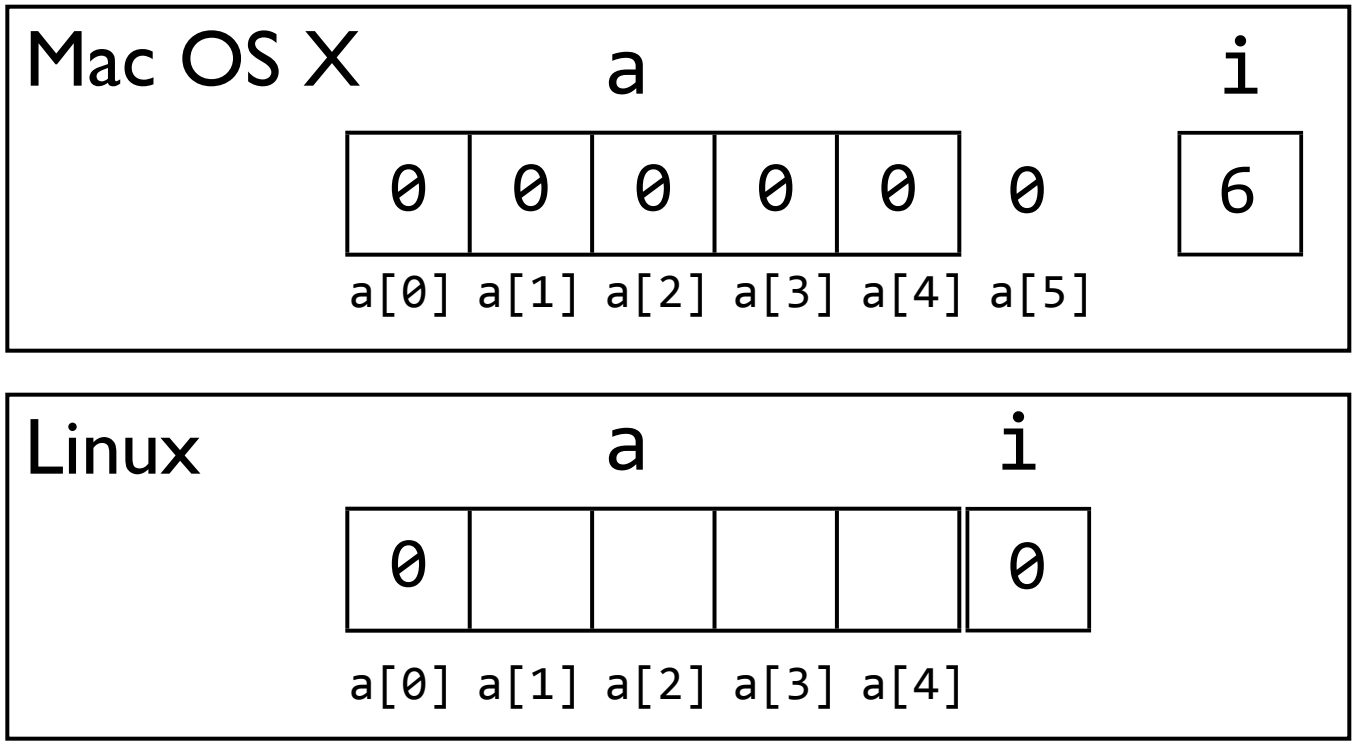



```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```

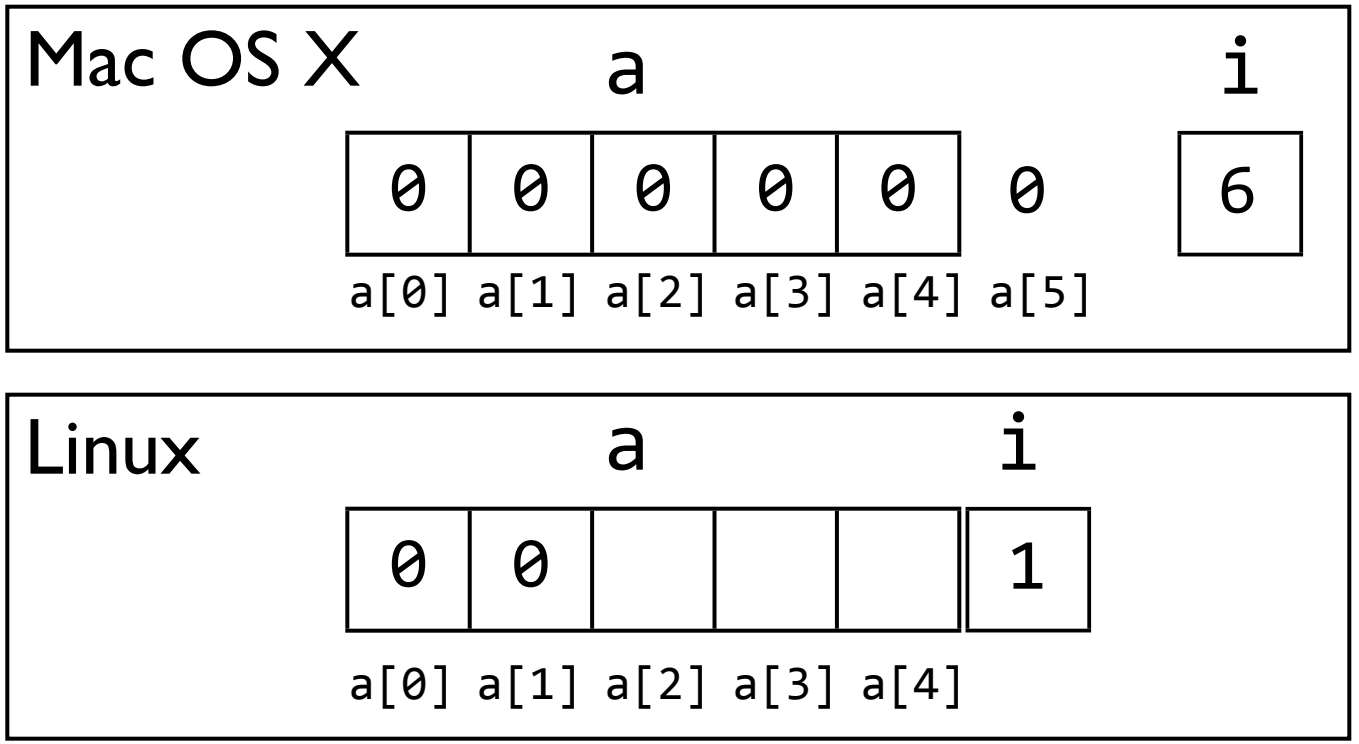
```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



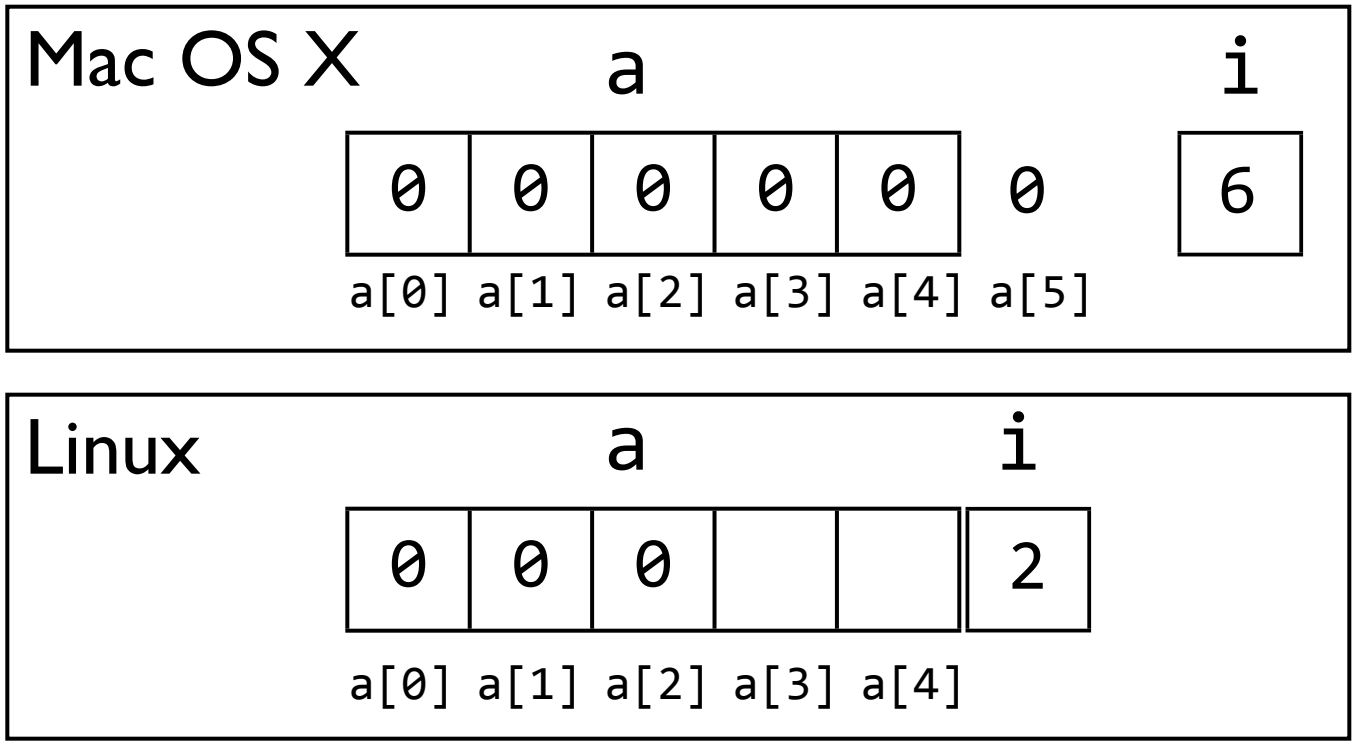
```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



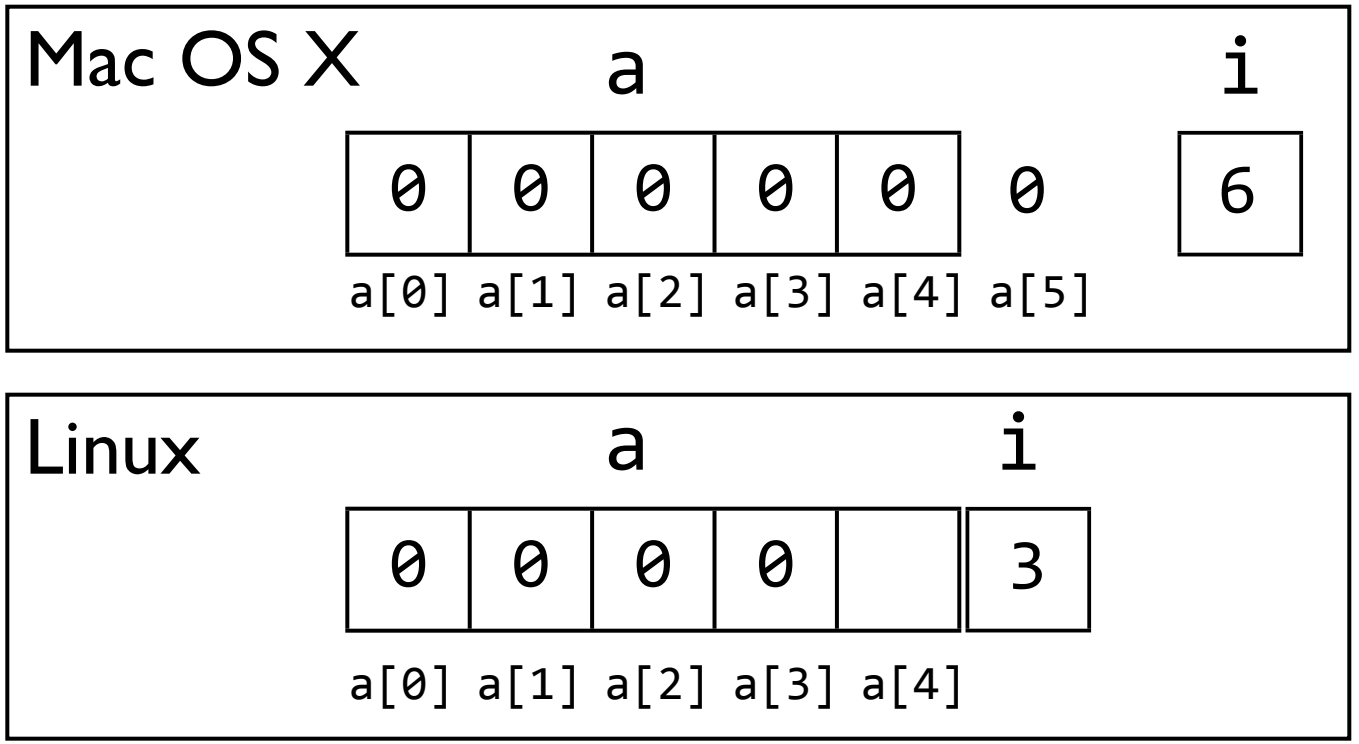
```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



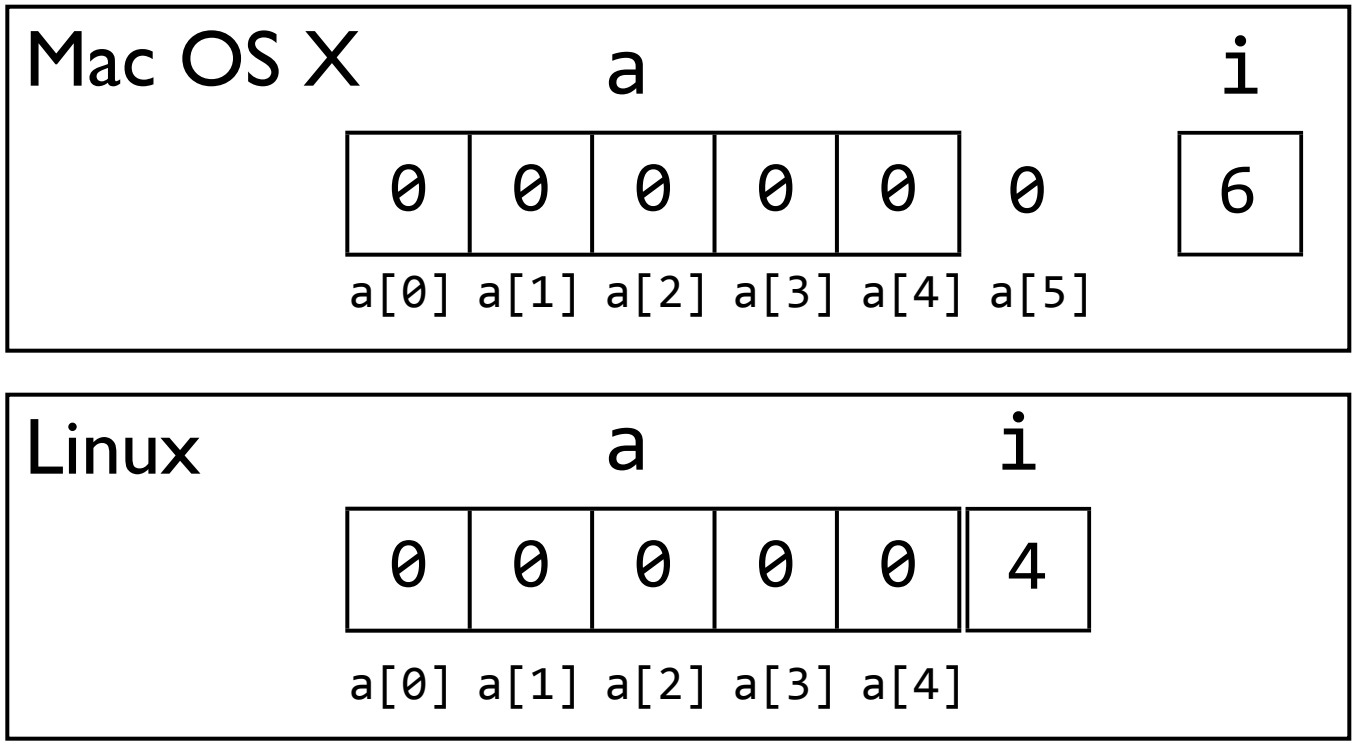
```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```

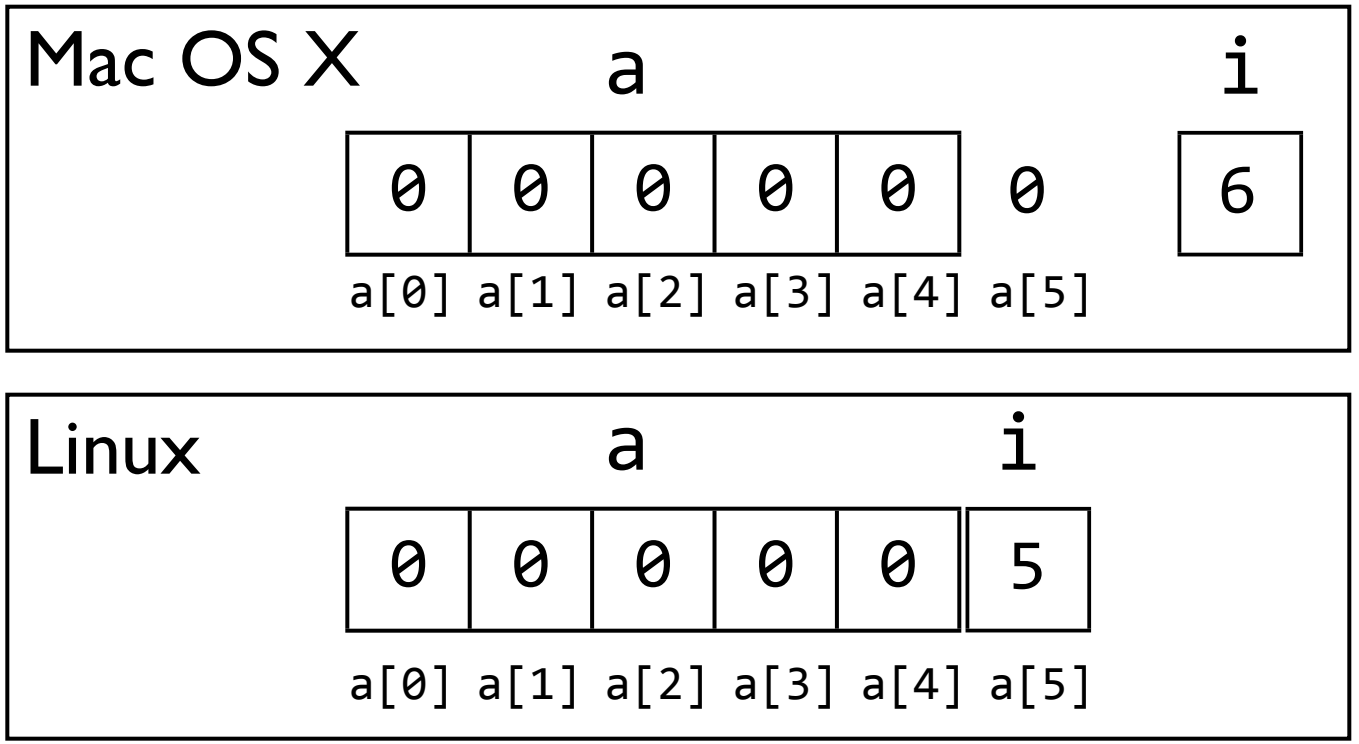


```
#define N 5
...

int a[N];
int i;
```

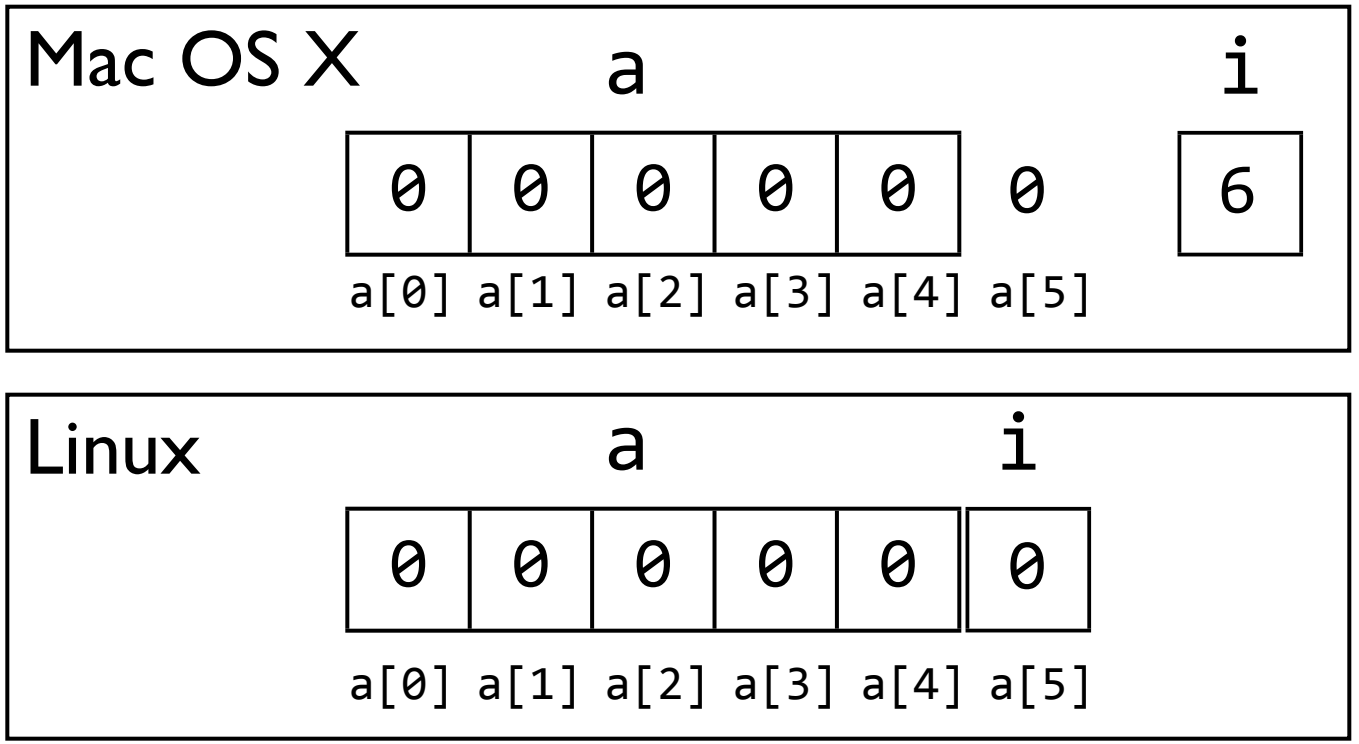



```
printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```

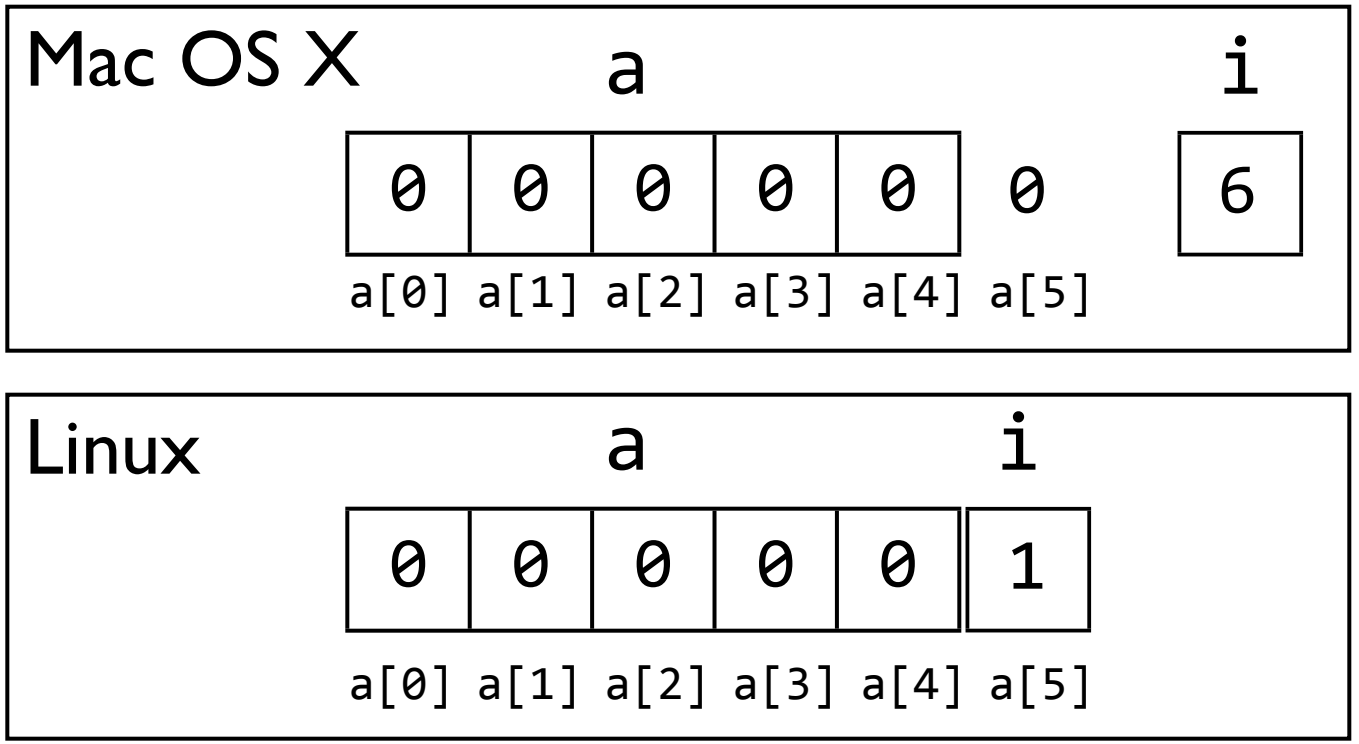
```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



```
#define N 5
...

int a[N];
int i;

printf("Beginning loop...\n");
for (i = 0; i <= N; i++)
{
    a[i] = 0;
}
printf("Ending loop...\n");
```



```
#define N 3
...
int first = 0;
int second = 0;
int third = 0;

for (int i = 0; i < N; i++)
{
    printf("Enter a number: ");
    scanf("%d",  ???  );
}
```

23
47
39

```
39 47 23
```

```
#define N 3
...
int a[N];
```

23
47
39

```
39 47 23
```

```
for (int i = 0; i < N; i++)
{
    printf("Enter a number: ");
    scanf("%d", &a[i] );
}
```

```
#define N 3
...
int a[N];
```

23
47
39

39 47 23

```
for (int i = 0; i < N; i++)
{
    printf("Enter a number: ");
    scanf("%d", &a[i] );
}
for (int i = N - 1; i >= 0; i-- )
{
    printf("%d ", a[i]);
}
```