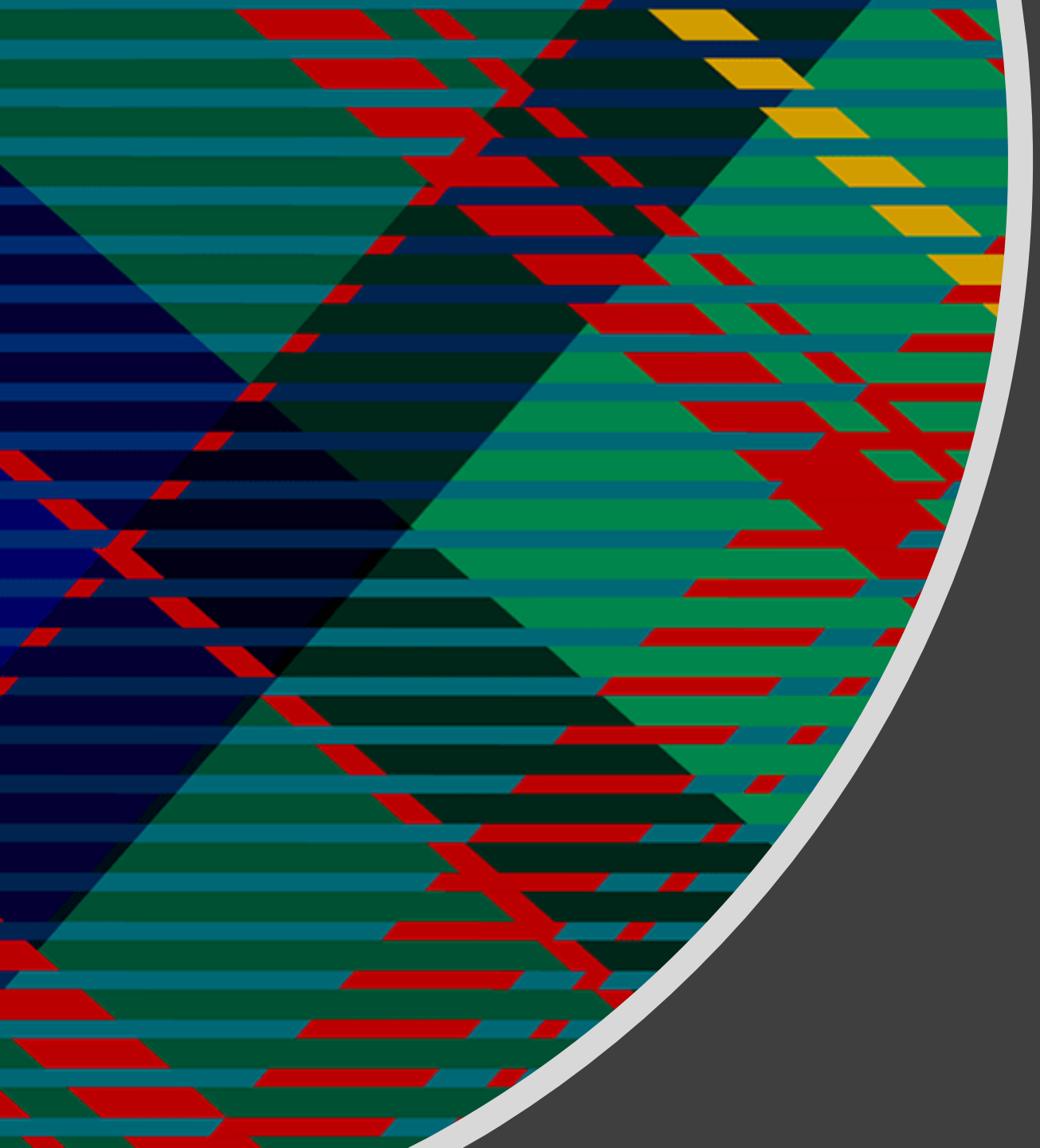


122
Arrays

Last time

Integers

- Representation (binary/hex/decimal)
- Modular arithmetic
- Two’s complement
- Integer Operators

Today

Arrays

- Simple memory model
- Array mechanics
- Aliasing
- Safety

Next

Search

Simple memory model

```
int ReLU(int x) {
  if (x <= 0)
    return 0;
  else
    return x;
}
int neuron(int x) {
  int w = 3;
  int b = 2;
  x = ReLU(w*x + b);
  return x;
}
int main() {
  int input = -5;
  int y = neuron(input);
  return y;
}
```

Arrays in C0

Card Question

```
int[] X = alloc_array(int, 3);
X[0] = 5;
X[1] = 6;
X[2] = 7;

int[] Y = X;
Y[2] = 9;
```

What does `X == Y` evaluate to?

A `true`

B `false`

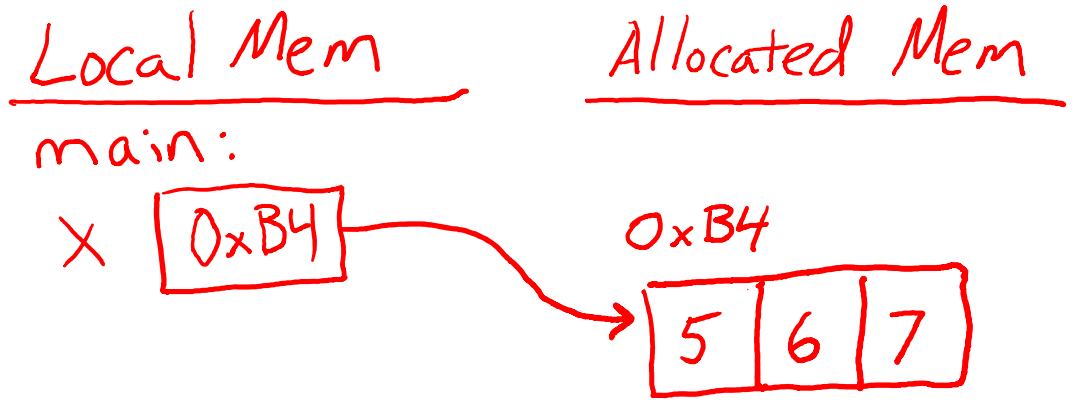
C Other

D I don't know

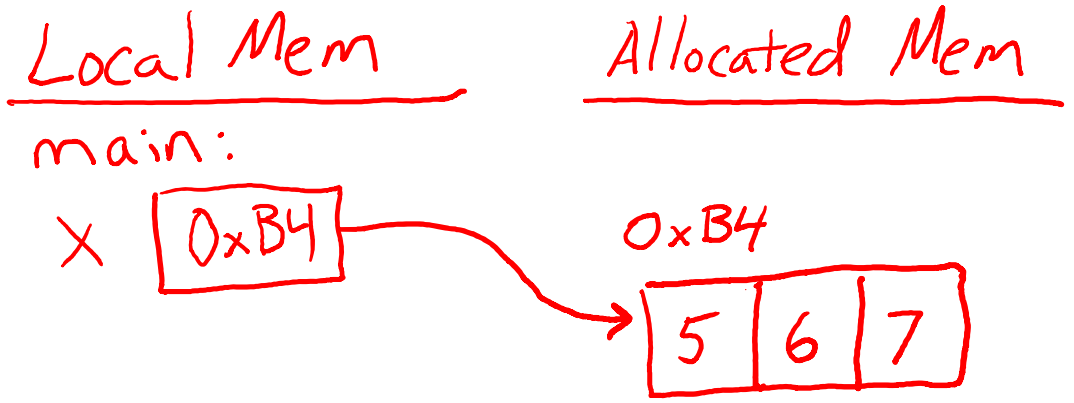
```
int[] X = alloc_array(int, 3);
X[0] = 5;
X[1] = 6;
X[2] = 7;

int[] Y = X;
Y[2] = 9;
X == Y;

int[] Z = alloc_array(int, 3);
Z[2] = 9;
X == Z;

X = alloc_array(int, 5);
Y = alloc_array(int, 2);
```

Copy array

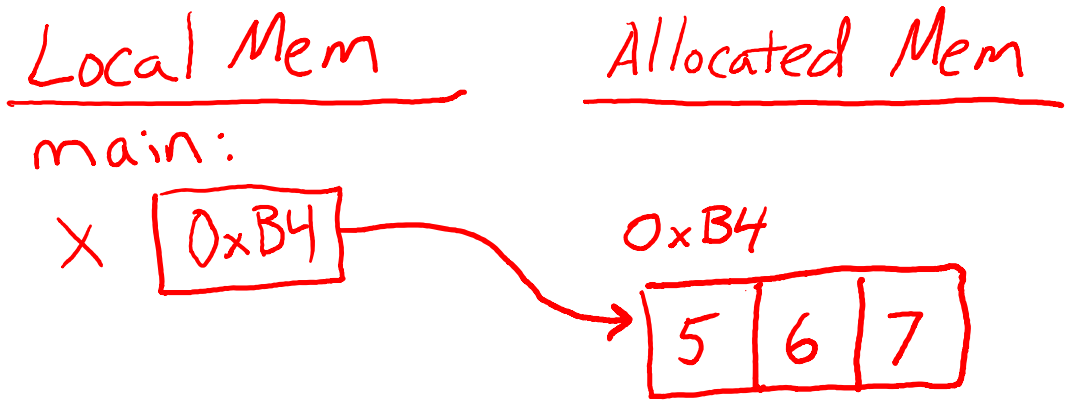
```
int[] copy_array(int[] A)
{
  return A;
}

int main()
{
  int[] X = alloc_array(int, 3);

  for (int i = 0; i < 3; i++) {
    X[i] = i+5; // A is [5, 6, 7]
  }
  int[] Y = copy_array(X);
  return 0;
}
```



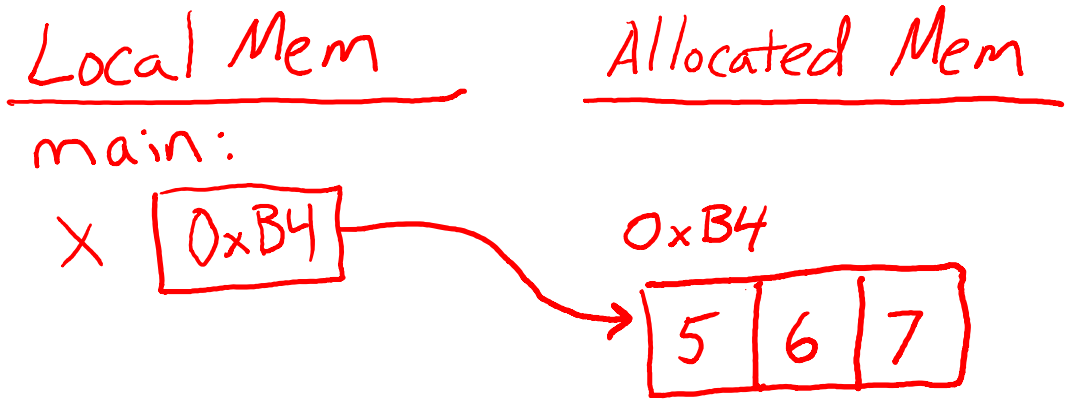
```
int[] Y = copy_array(X, 3);

-------------------------------

int[] copy_array(int[] A)
{
  int[] B = alloc_array(int, ???);
  ...

}
```



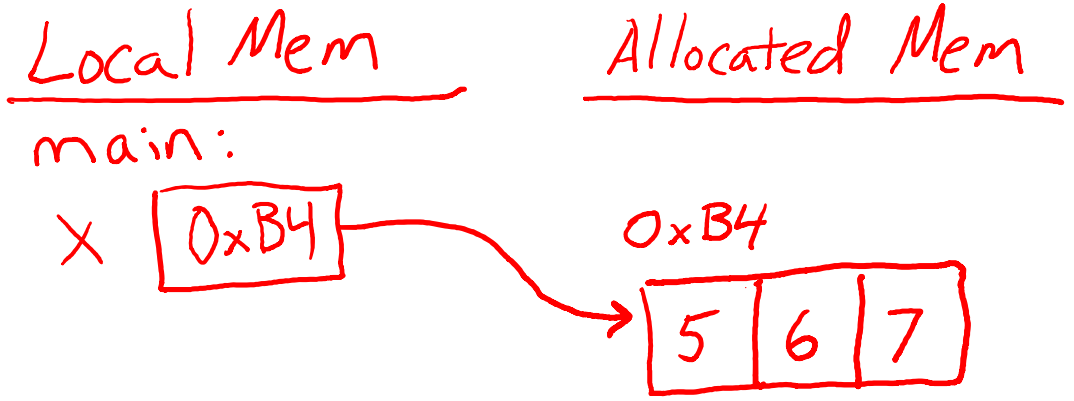
```
int[] Y = copy_array(X, 3);

-------------------------------

int[] copy_array(int[] A, int n)
{
  int[] B = alloc_array(int, n);
  ...

}
```



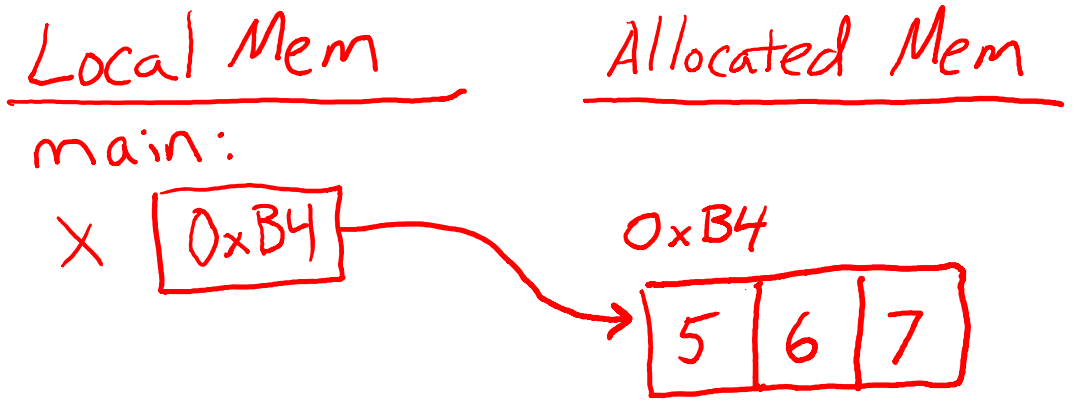
```
int[] Y = copy_array(X, 3);

-------------------------------

int[] copy_array(int[] A, int n)
//@requires n == \length(A);
{
  int[] B = alloc_array(int, n);
  ...

}
```



```
int[] Y = copy_array(X, 3);

-------------------------------

int[] copy_array(int[] A, int n)
//@requires n == \length(A);
{
  int[] B = alloc_array(int, n);
  B=A;
  return B;
}
```



```
int[] Y = copy_array(X, 3);

-------------------------------


int[] copy_array(int[] A, int n)
//@requires n == \length(A);
{
  int[] B = alloc_array(int, n);
  for (int i = 0; i < n; i++) {
    B[i] = A[i];
  }
  return B;
}
```



Safety

```
1   int[] copy_array(int[] A, int n)
2   //@requires n == \length(A);
3   {
4     int[] B = alloc_array(int, n);
5     for (int i = 0; i < n; i++){
6     {
7       B[i] = A[i];
8     }
9     return B;
10  }
```

Is `A[i]` safe?

1) i < \length(A) ??

   \length(A) == n *by line 2*

   i < n *by line 5 loop guard*

2) i >=0 ??

Safety

```
1   int[] copy_array(int[] A, int n)
2   //@requires n == \length(A);
3   {
4     int[] B = alloc_array(int, n);
5     for (int i = 0; i < n; i++){
6     //@loop_invariant i >= 0;
7     {
8       B[i] = A[i];
9     }
10    return B;
11  }
```

Is `A[i]` safe?

1) i < \length(A) ??

   \length(A) == n *by line 2*

   i < n *by line 5 loop guard*

2) i >=0 ??

   loop invariant on line 6

Is `B[i]` safe?

   \length(B) == n *by line 4*

Safety

```
1   int[] copy_array(int[] A, int n)
2   //@requires n == \length(A);
3   {
4     int[] B = alloc_array(int, n);
5     for (int i = 0; i < n; i++){
6     //@loop_invariant i >= 0;
7     {
8       B[i] = A[i];
9     }
10    return B;
11  }
```

Is `A[i]` safe?

1) i < \length(A)
2) i >=0

loop invariant on line 6

Is the loop invariant valid?

1) INIT:
2) PRES: assume i > = 0,

show i' >= 0

Safety

Is line 20 safe?

Is line 21 safe?

```
1   int[] copy_array(int[] A, int n)
2   //@requires n == \length(A);
3
…
15  int main() {
16    int[] X = alloc_array(int, 3);
17    X[0] = 5;
18    X[1] = 6;
19    X[2] = 7;
20    int[] Y = copy_array(int, 3);
21    int[] Z = copy_array(int, 3);
22    return B;
23  }
```

Safety

Is line 20 safe?

Is line 21 safe?

```
1   int[] copy_array(int[] A, int n)
2   //@requires n == \length(A);
3   //@ensures \length(\result) == n;
…
15  int main() {
16    int[] X = alloc_array(int, 3);
17    X[0] = 5;
18    X[1] = 6;
19    X[2] = 7;
20    int[] Y = copy_array(int, 3);
21    int[] Z = copy_array(int, 3);
22    return B;
23  }
```

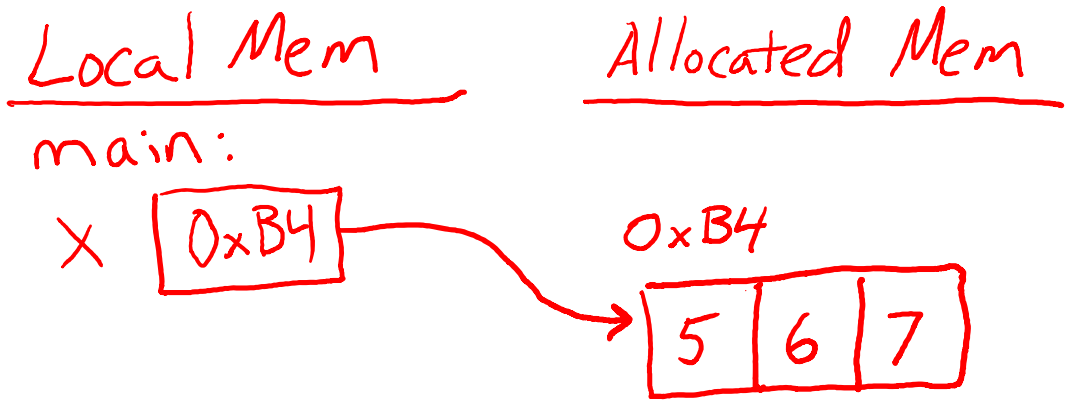
Correctness

Is `copy_array` correct?

```
1  int[] copy_array(int[] A, int n)
2  //@requires n == \length(A);
3  //@ensures \length(\result) == n;
4  {
5    int[] B = alloc_array(int, n);
6    for (int i = 0; i < n; i++){
7    //@loop_invariant i >= 0;
8    {
9      B[i] = A[i];
10   }
11   return B;
12 }
```

Card question

```
int[] Y = copy_array(X, 3);
```




```
int[] copy_array(int[] A, int n)
{
  int[] B = alloc_array(int, n);
  for (int i = 0; i < n; i++){
  {
    B[i] = A[i];
  }
  A = alloc_array(int, 42);
  return B;
}
```

```
int[] Y = copy_array(X, 3);
```

What is the length of X?

A	3
B	42
C	Error
D	Other

Array addresses

```
1   int[] copy_array(int[] A, int n)
2   //@requires n == \length(A);
3   //@ensures \length(\result) == n;
4   {
5     int[] B = alloc_array(int, n);
6     for (int i = 0; i < n; i++){
7     //@loop_invariant i >= 0;
8     {
9       B[i] = A[i];
10    }
11    A = alloc_array(int, 42);
12    return B;
13  }
```