



UNIVERSITY OF
RICHMOND

CMSC 240 Lecture 5

CMSC 240 Software Systems Development
Fall 2024

Today

- Memory
 - Pointers
 - Garbage Values
 - Memory Layout
- In-class exercise



Today

- Memory

- Pointers

- Garbage Values

- Memory Layout

- In-class exercise



char 1 byte

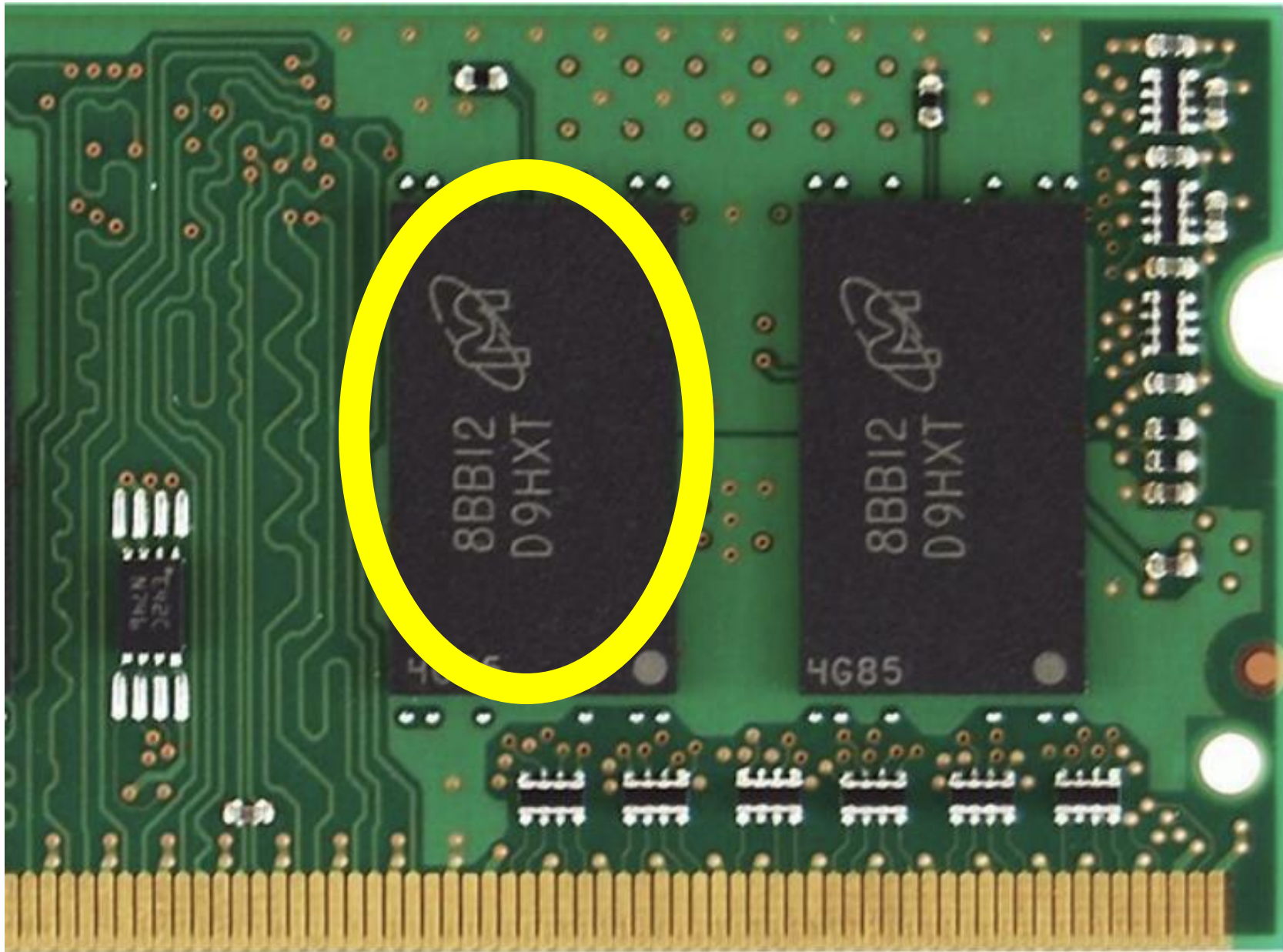
int 4 bytes

long 8 bytes

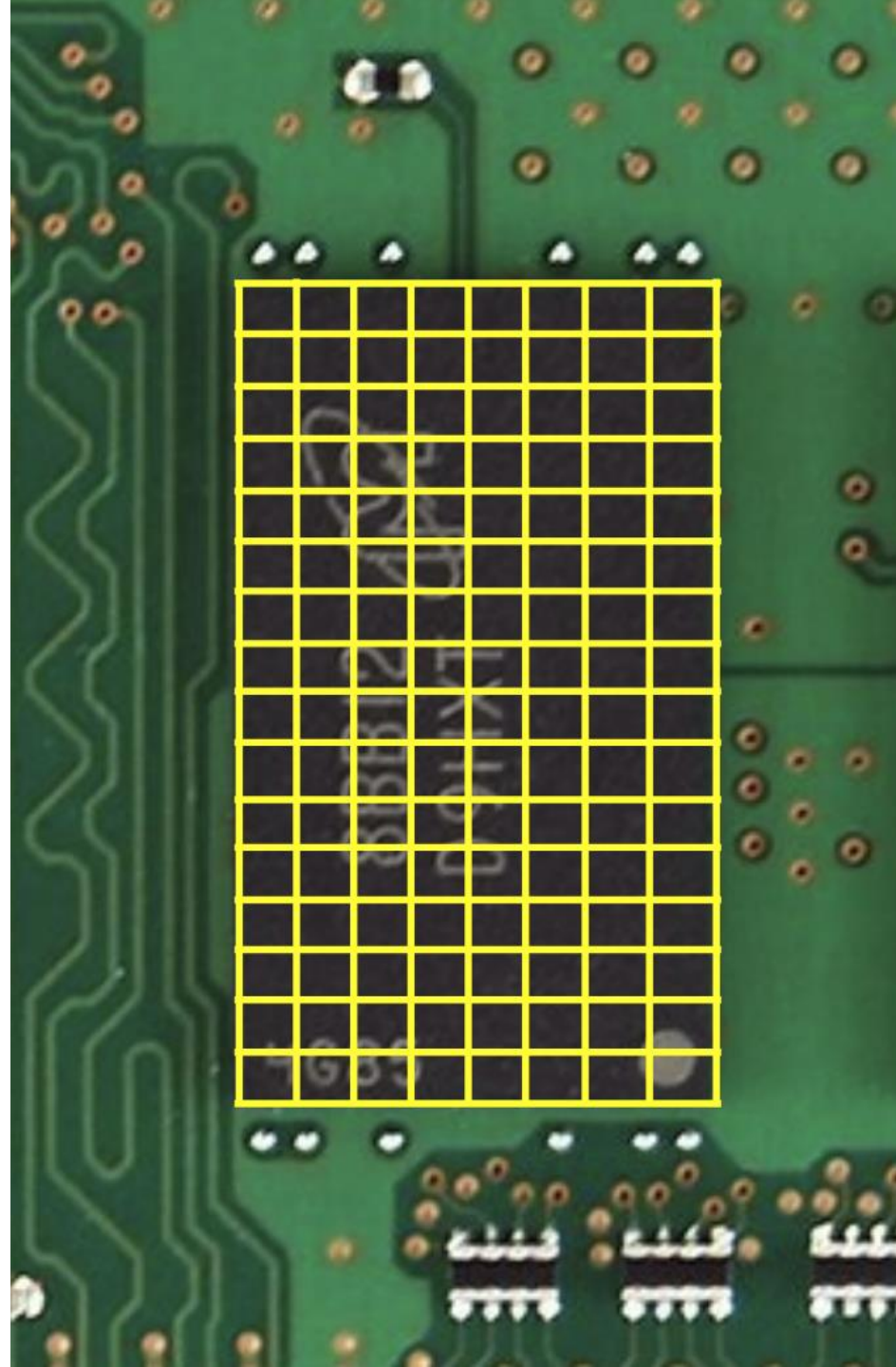
float 4 bytes

double 8 bytes

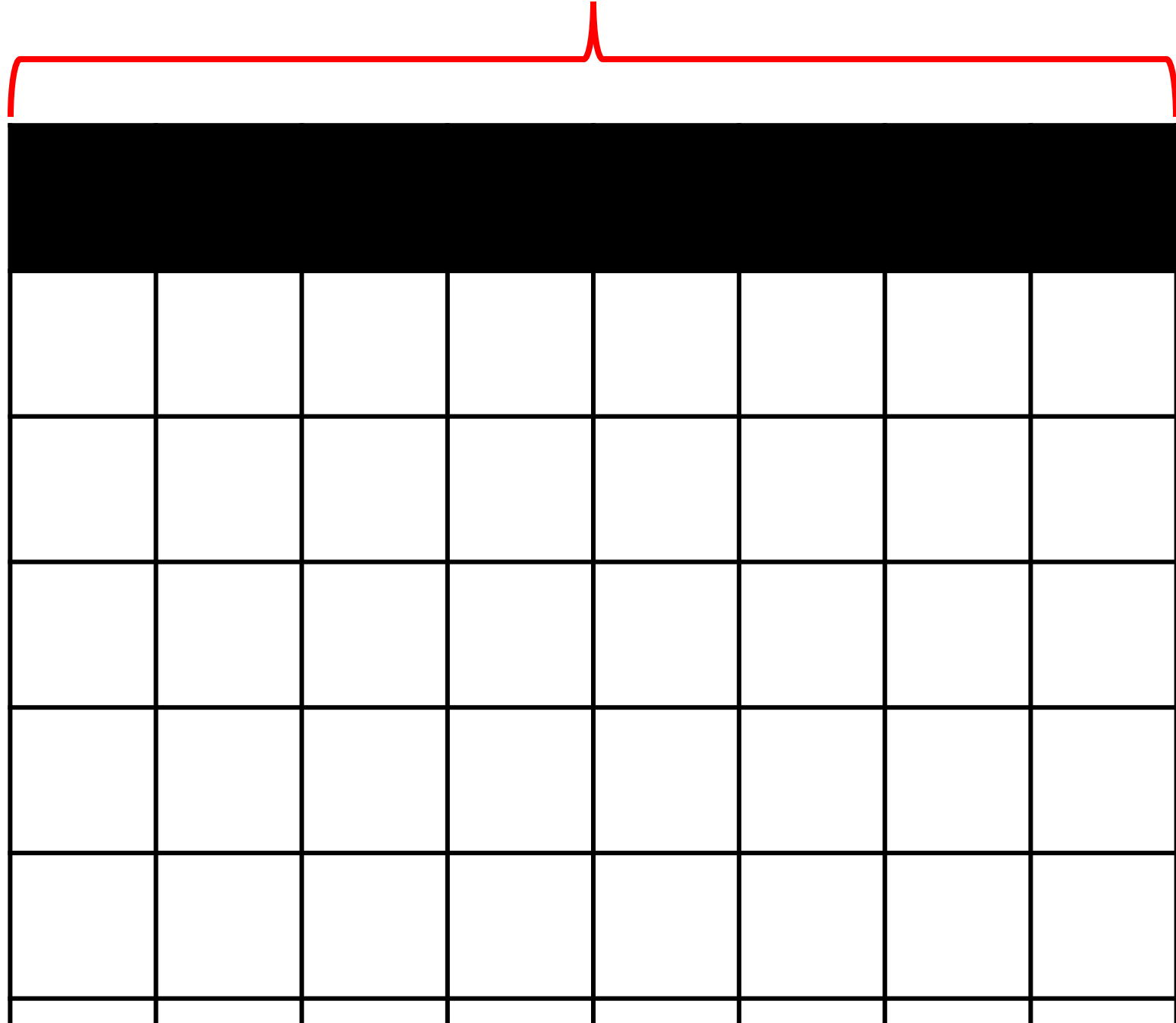
string ?







double 8 bytes



```
int age1 = 42;
```

```
int age2 = 27;
```

```
int age3 = 19;
```




0 1

0 1 2 3 4 5 6 7 8 9

base-16

0 1 2 3 4 5 6 7 8 9 A B C D E F

Hexadecimal

2^7 2^6 2^5 2^4 2^3 2^2 2^1 2^0

1 1 1 1 1 1 1 1

128 64 32 16 8 4 2 1

1 1 1 1 1 1 1 1

128 64 32 16 8 4 2 1

1 1 1 1 1 1 1 1

$(128 \times 1) + (64 \times 1) + (32 \times 1) + (16 \times 1) + (8 \times 1) + (4 \times 1) + (2 \times 1) + (1 \times 1)$

128 64 32 16 8 4 2 1

1 1 1 1 1 1 1 1

255

16^1 16^0

**#**

16

1

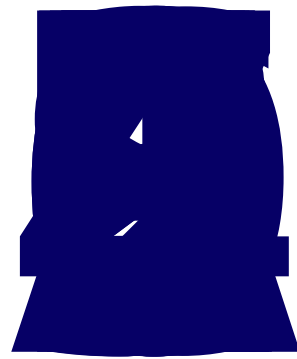
#

#

16



1



16

1

F

F

16

1

F

F

$$(16 \times F) + (1 \times F)$$

16

1

F

F

$$(16 \times 15) + (1 \times 15)$$

16

1

F

F

240

+

15

16

1

F

F

255

128 64 32 16 8 4 2 1

1 1 1 1 1 1 1 1

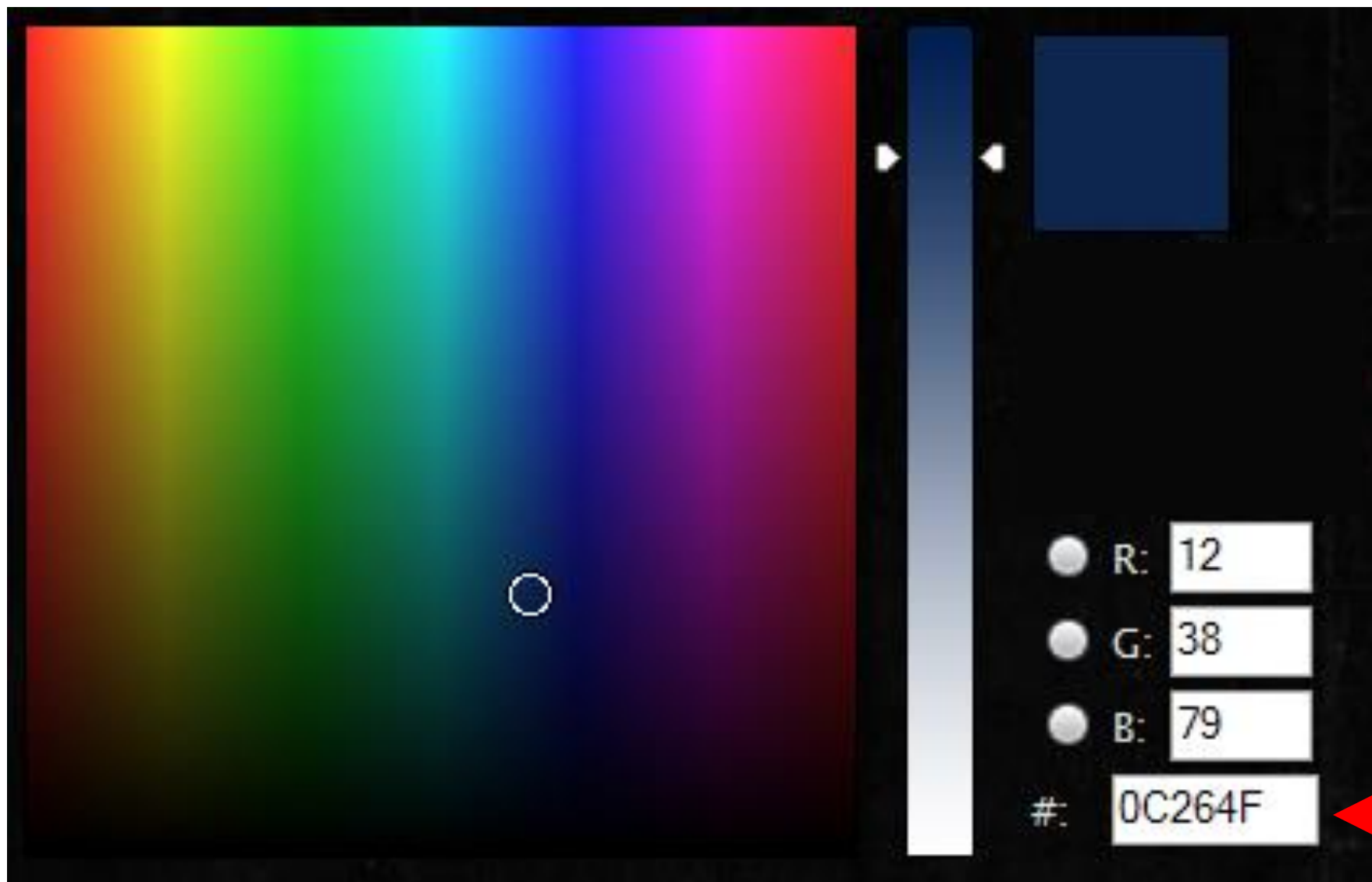
255

1 1 1 1 1 1 1 1

1 1 1 1 1 1 1 1

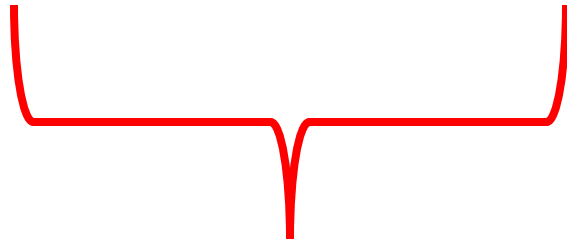
F

F



0	1	2	3	4	5	6	7
8	9	A	B	C	D	E	F
10	11	12	13	14	15	16	17
18	19	1A	1B	1C	1D	1E	1F
20	21	22	23	24	25	26	27
28	29	2A	2B	2C	2D	2E	2F
30	31	32	33	34	35	36	37

0x F F



C++ convention that means
the following number is a
hexadecimal (base-16)

0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17
0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F
0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27
0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F
0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37

```
int var = 17;
```

				17			
				var			

				0x123	17		
					var		

Ask a question



Today

- ~~Memory~~

- **Pointers**

- Garbage Values

- Memory Layout

- In-class exercise



&

“address of”

*

“contents of”

A **pointer** is an object that holds an address value.

```
int var = 17;
```

```
int* ptr = &var; // ptr holds the address of var
```

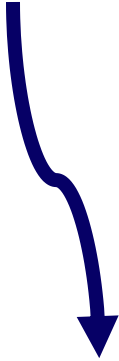
This new type is called an **"int pointer"** and is used to hold the address of an integer variable

The **"address of"** the integer `var`

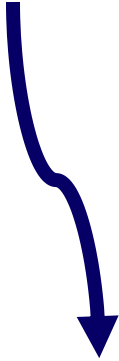
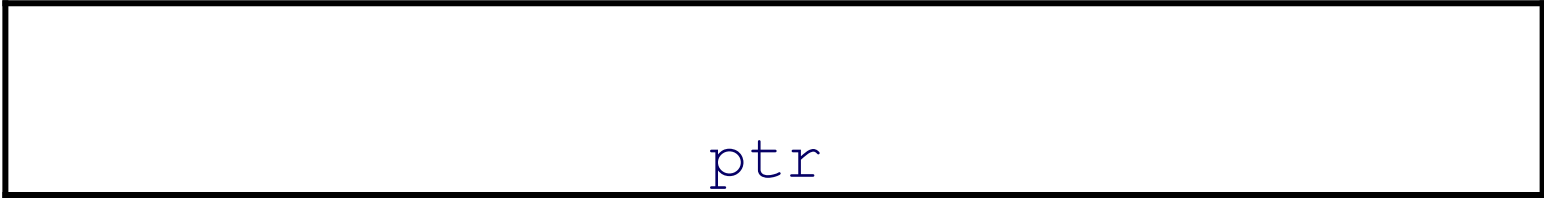
				0x16D01345			
					17		
					var		

0x16D01345 ptr							
				0x16D01345	17 var		

0x16D01345
ptr



0x16D01345
17
var



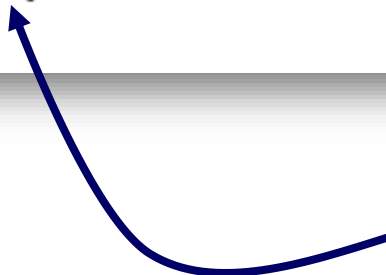
```
int var = 17;
```

```
int* ptr = &var; // ptr holds the address of var
```

```
int anotherVar = *ptr;
```



17



The "**contents of**" the
pointer `ptr`
(i.e., the value stored
at the address)


```
int var = 17;
int* ptr = &var; // ptr holds the address of var

cout << "var == " << var << endl; // #1
cout << "'address of' var == " << &var << endl; // #2

cout << "ptr == " << ptr << endl; // #3
cout << "'contents of' ptr == " << *ptr << endl; // #4

cout << "'contents of' the 'address of' var == " << *&var << endl; // #5
```



```
int var = 17;
int* ptr = &var; // ptr holds the address of var

cout << "var == " << var << endl; // #1
cout << "'address of' var == " << &var << endl; // #2

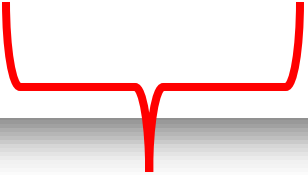
cout << "ptr == " << ptr << endl; // #3
cout << "'contents of' ptr == " << *ptr << endl; // #4

cout << "'contents of' the 'address of' var == " << *&var << endl; // #5
```

```
var == 17
'address of' var == 0x16af27454
ptr == 0x16af27454
'contents of' ptr == 17
'contents of' the 'address of' var == 17
```

```
double pi = 3.14159;
```

```
double* ptr = &pi; // ptr holds the address of pi
```



This new type is called an
“**double pointer**” and is
used to hold the address of
a variable of type double



The “**Address of**”
the double `pi`

0x45FA1379

3.14159

pi

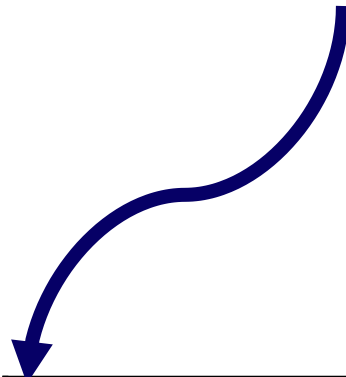
--	--	--	--	--	--	--	--

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

0x45FA1379 ptr							
0x45FA1379 3.14159 pi							

0x45FA1379

ptr



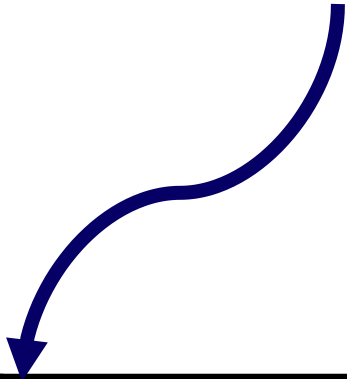
0x45FA1379

3.14159

pi

ptr

3.14159
pi



MAN, I SUCK AT THIS GAME.
CAN YOU GIVE ME
A FEW POINTERS?

0x3A28213A
0x6339392C,
0x7363682E.

I HATE YOU.



Ask a question



Today

- ~~Memory~~

- ~~Pointers~~

- **Garbage Values**

- Memory Layout

- In-class exercise



```
#include <iostream>
using namespace std;
```

```
int main()
{
```

```
    int var1;
    int var2;
    int var3;
    int var4;
    int var5;
```

```
var1 == 1
var2 == 71282756
var3 == 1
var4 == 71319552
var5 == 1
```

```
    cout << "var1 == " << var1 << endl;
    cout << "var2 == " << var2 << endl;
    cout << "var3 == " << var3 << endl;
    cout << "var4 == " << var4 << endl;
    cout << "var5 == " << var5 << endl;
```

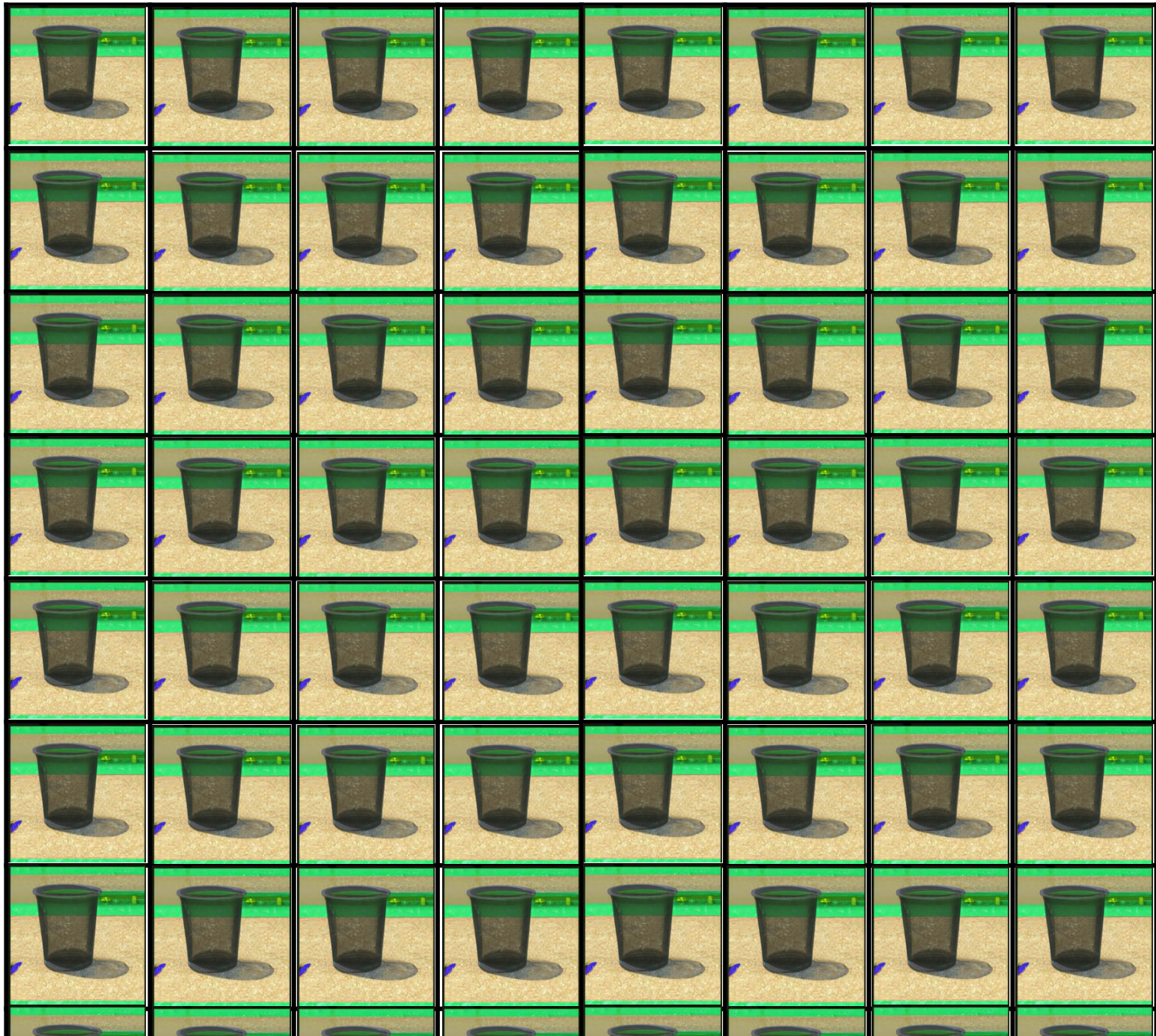
```
}
```

```
array<int, 10> numbers = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
```

```
cout << "numbers[323] == " << numbers[323] << endl; // Clearly out of array bounds!!
```

```
numbers[323] == 1966029422
```

0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17
0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F
0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27
0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F
0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37



Today

- ~~Memory~~

- ~~Pointers~~

- ~~Garbage Values~~

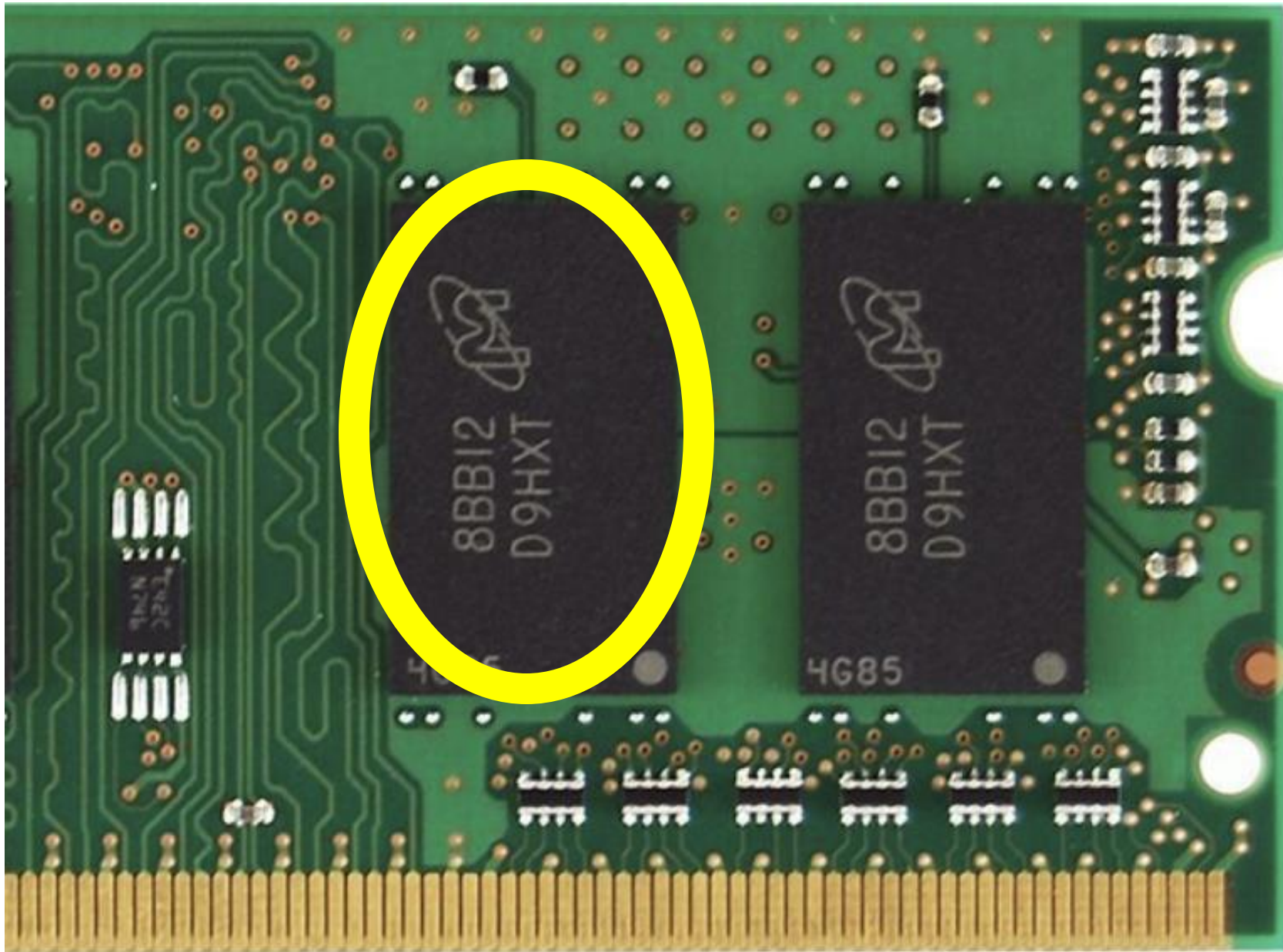
- Memory Layout

- In-class exercise

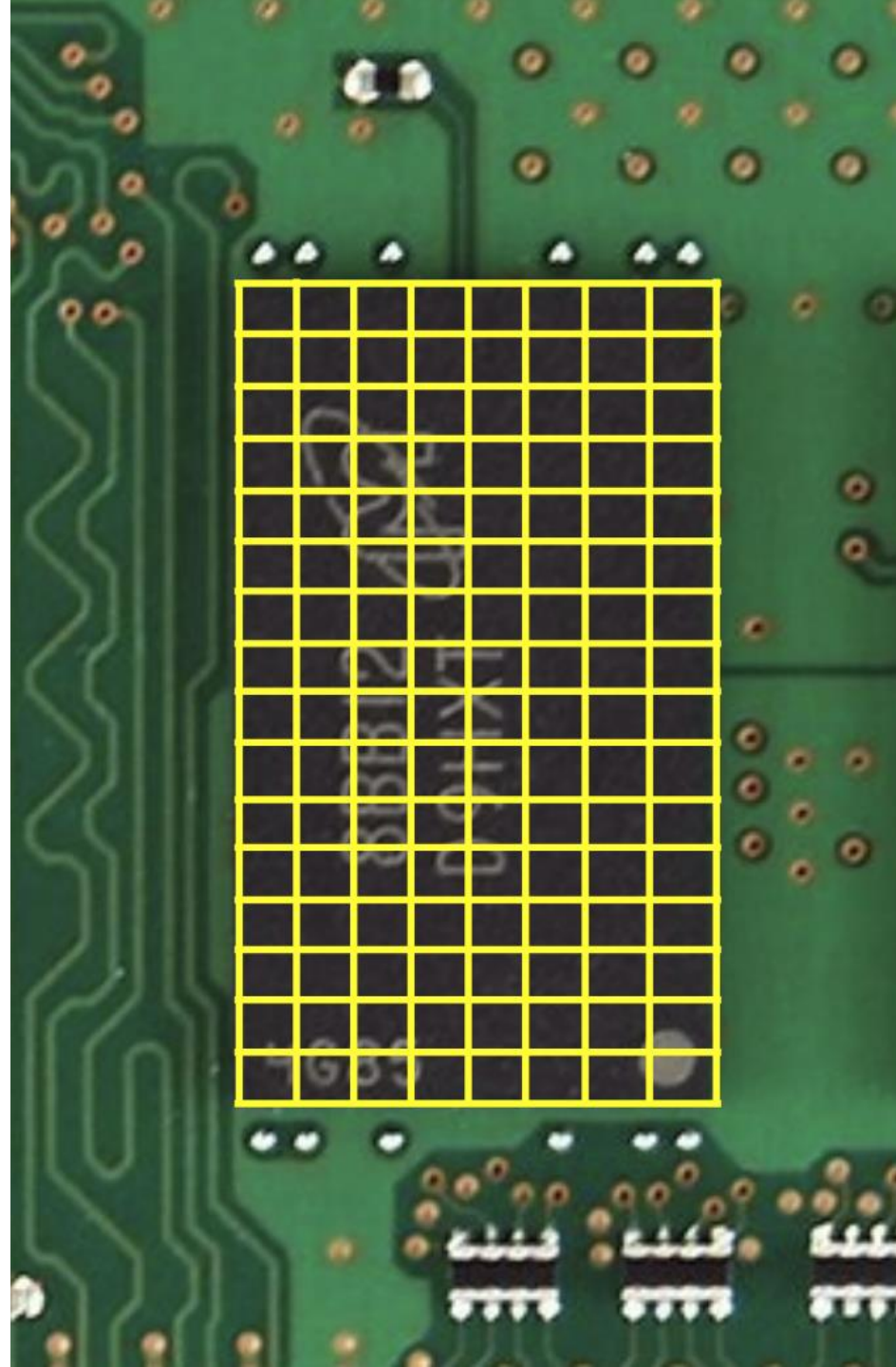


```
// Swap two int values.
void swap(int a, int b)
{
    int temp = a; // store a in temp
    a = b;         // put b into a
    b = temp;     // put temp a into b
}

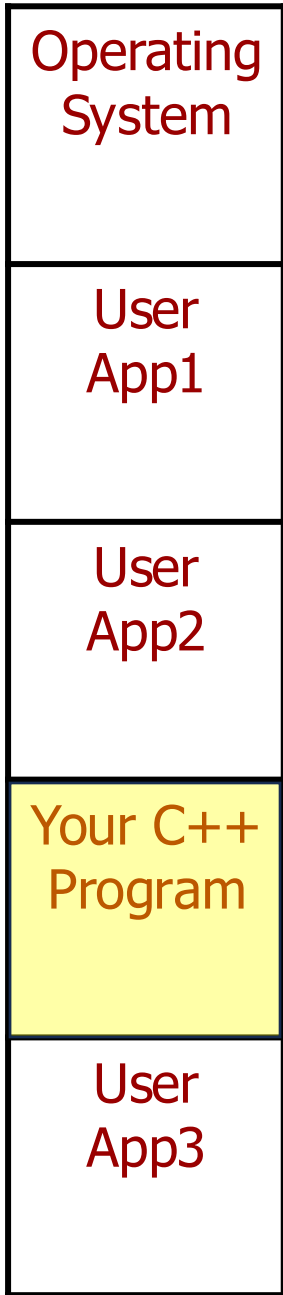
int main()
{
    int x = 12;
    int y = 33;
    swap(x, y);
    cout << "x == " << x << "   y == " << y << endl; // ?
    return 0;
}
```





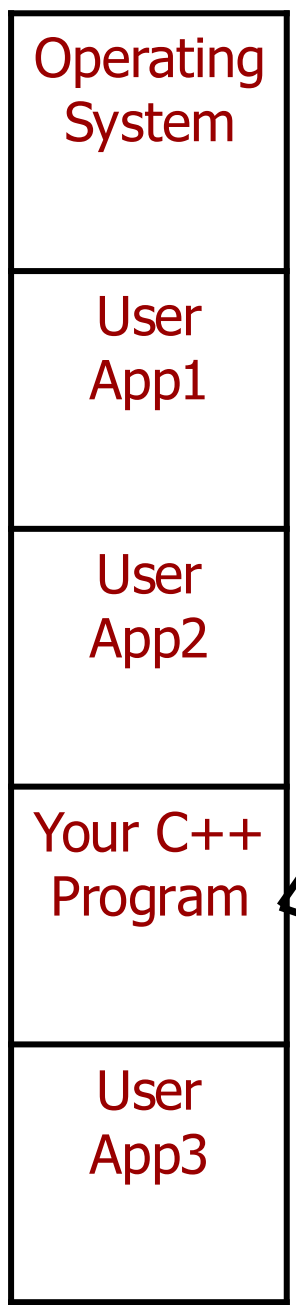
0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17
0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F
0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27
0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F
0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37
0x38	0x39	0x3A	0x3B	0x3C	0x3D	0x3E	0x3F
0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47
0x48	0x49	0x4A	0x4B	0x4C	0x4D	0x4E	0x4F



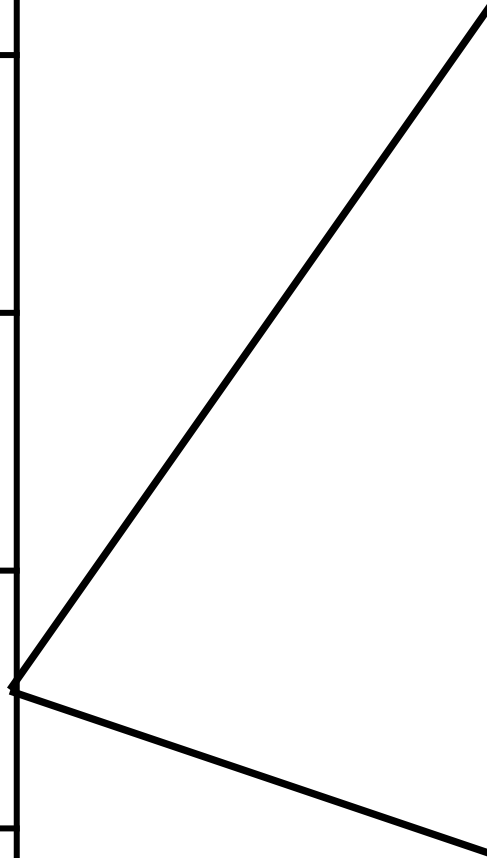
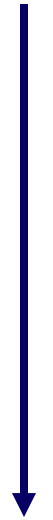
All of
main
memory



All of
main
memory

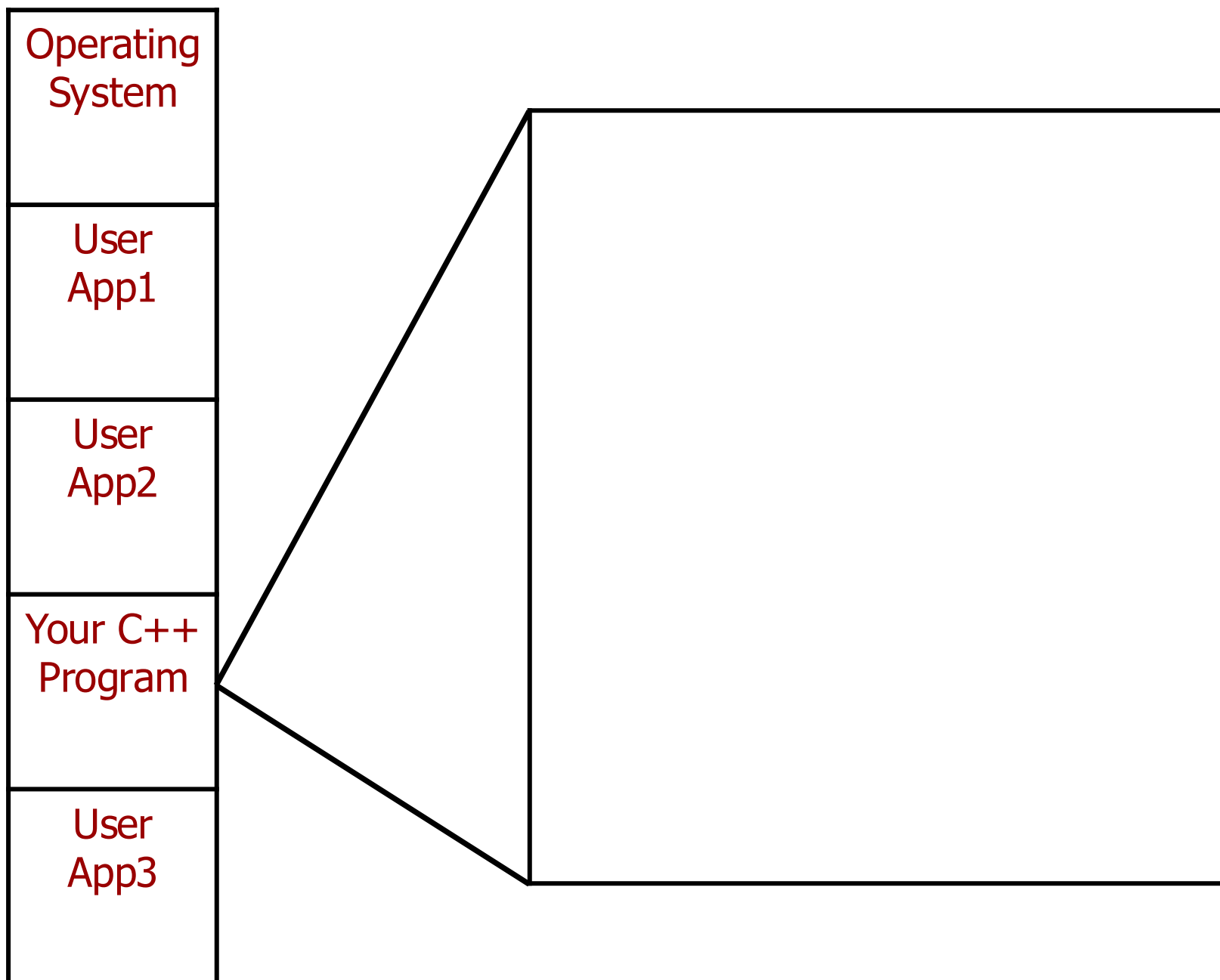


0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7
0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17
0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F
0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27
0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F
0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37
0x38	0x39	0x3A	0x3B	0x3C	0x3D	0x3E	0x3F
0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47
0x48	0x49	0x4A	0x4B	0x4C	0x4D	0x4E	0x4F

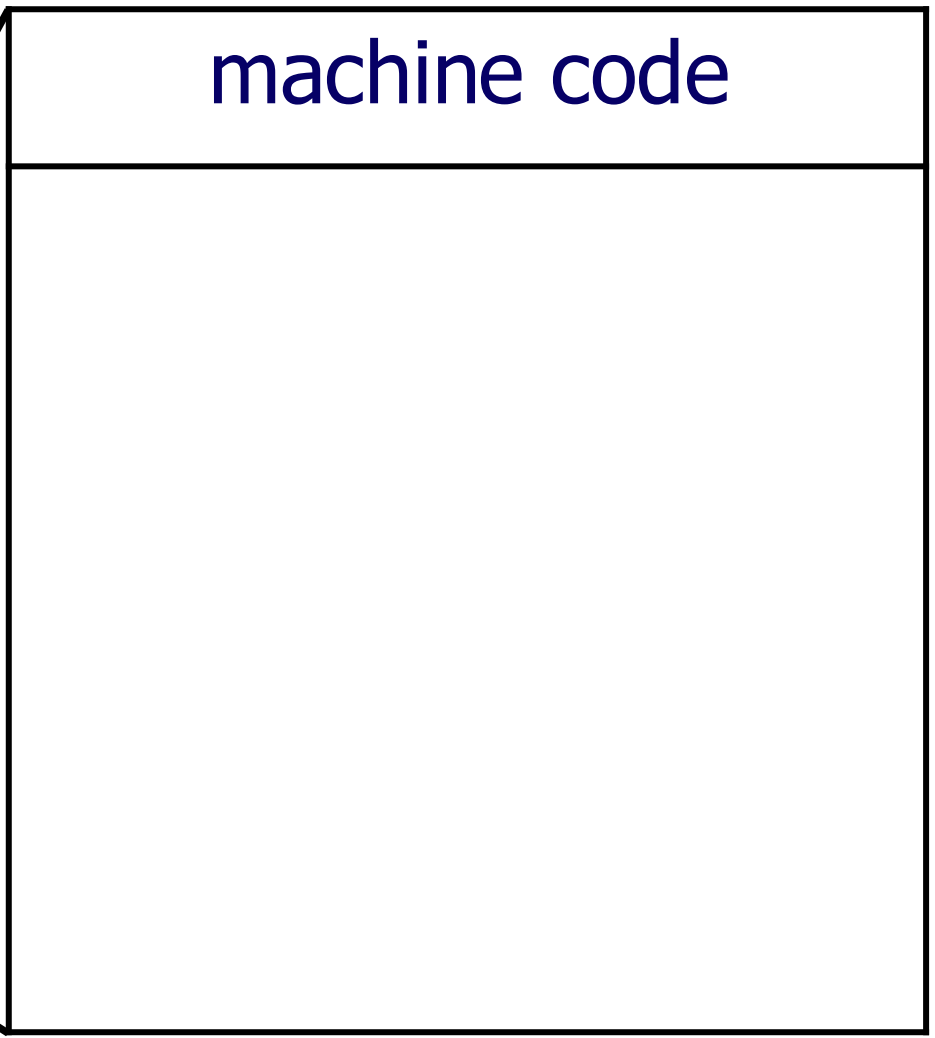
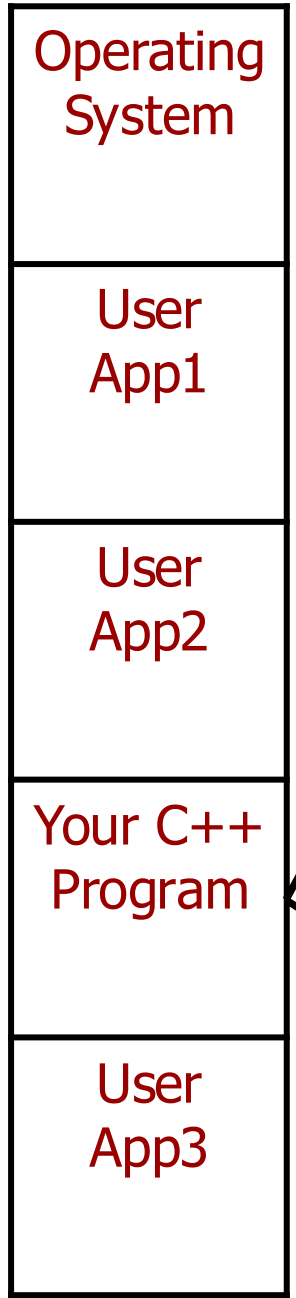




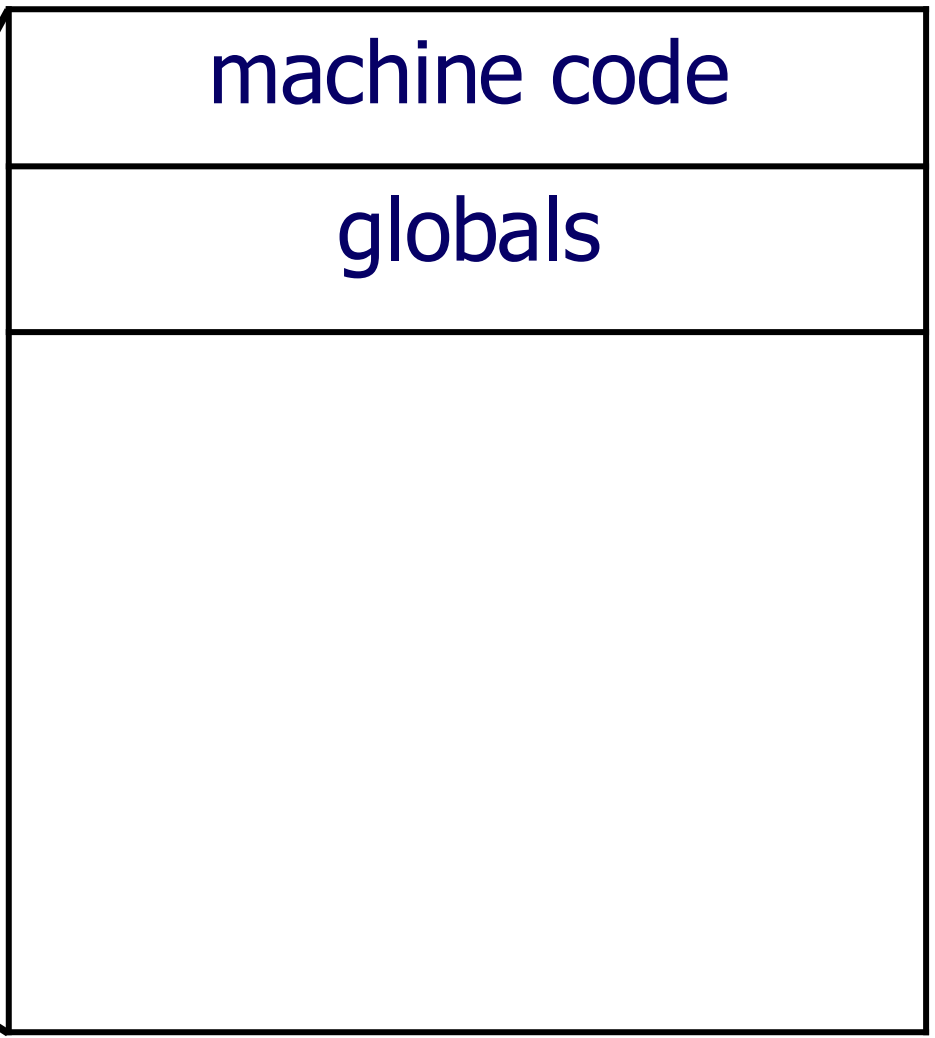
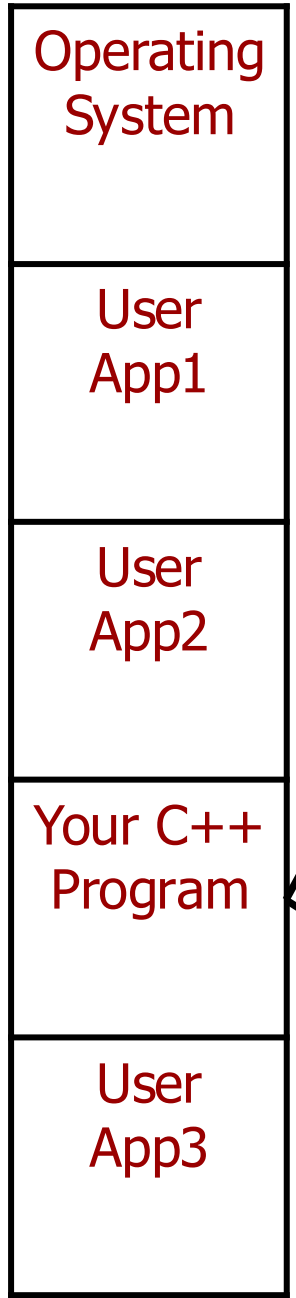
↑
All of
main
memory
↓



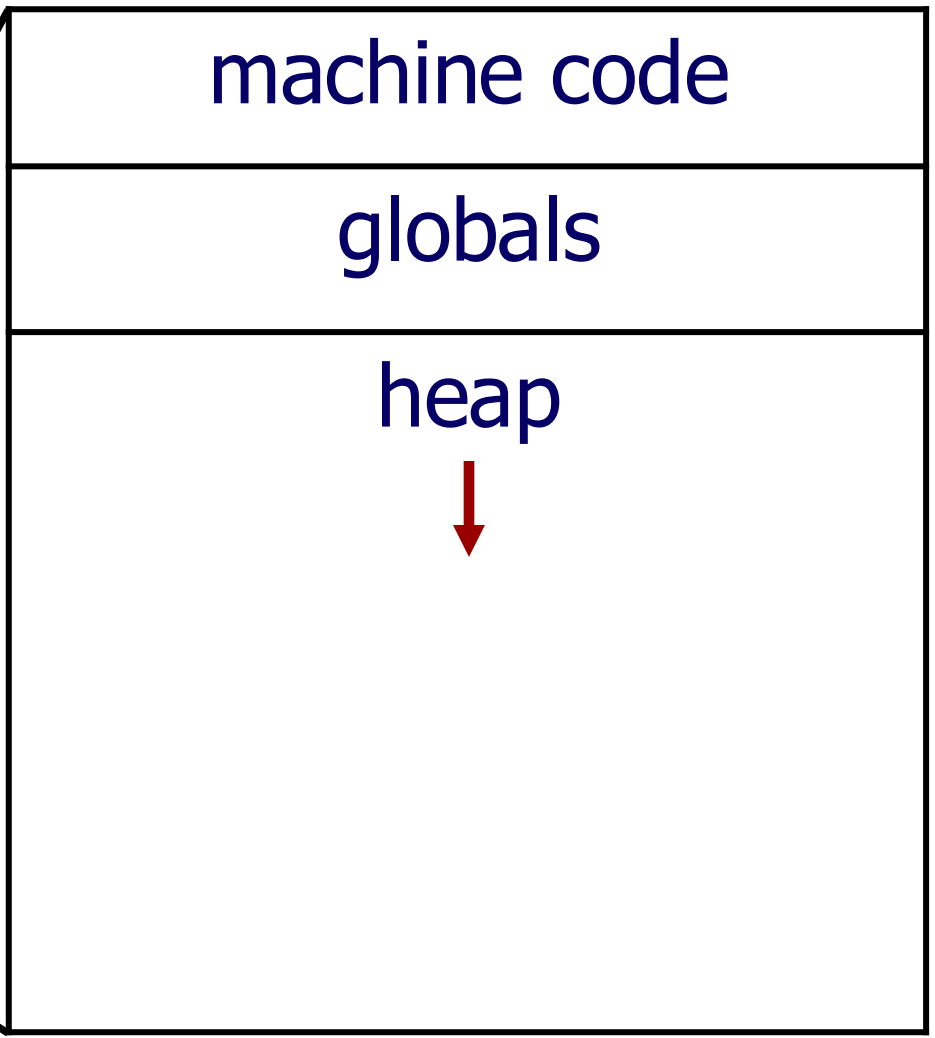
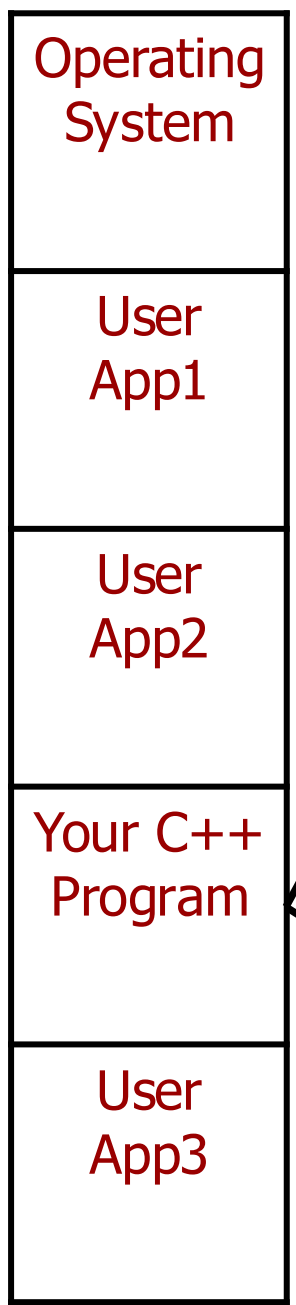
All of
main
memory

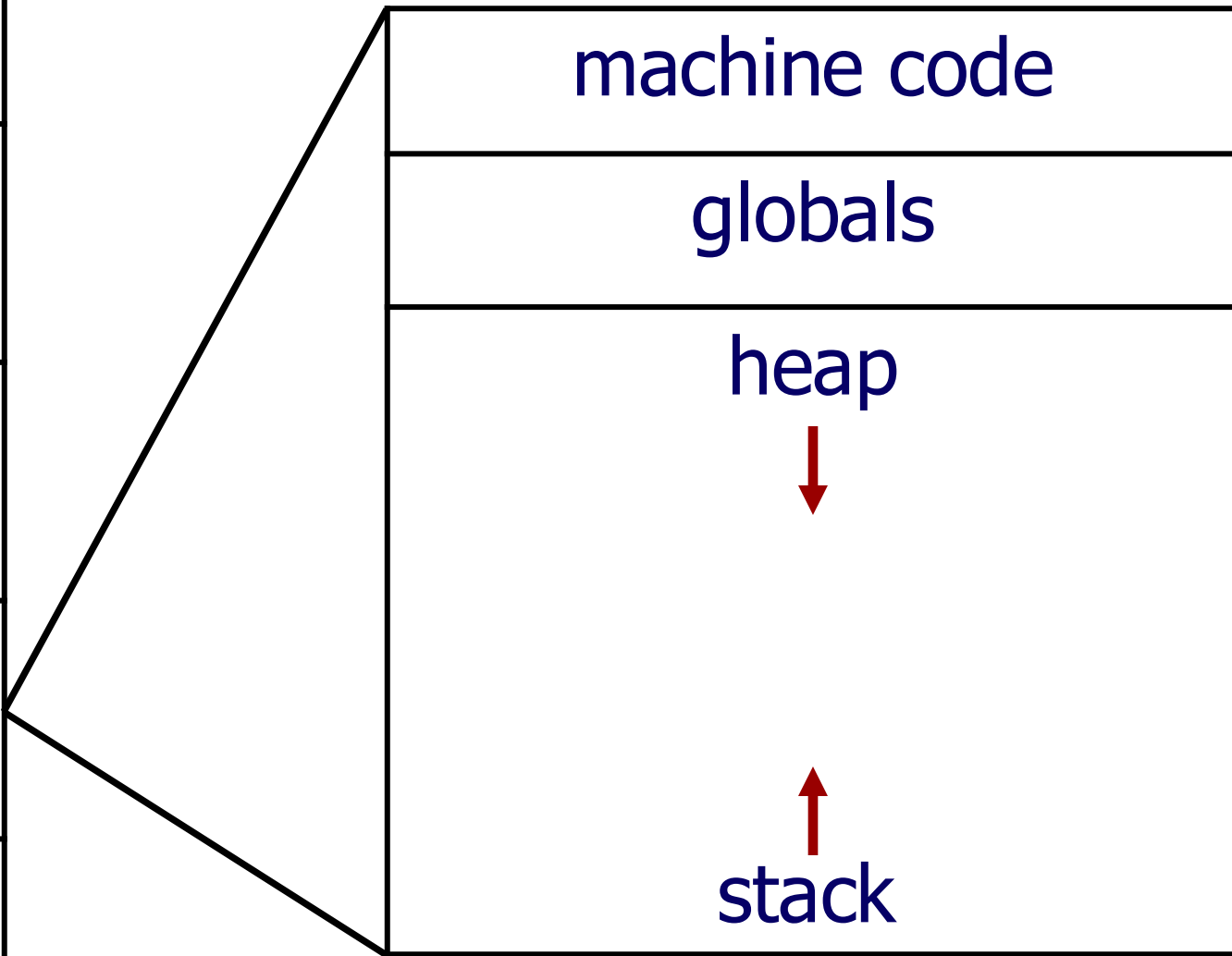
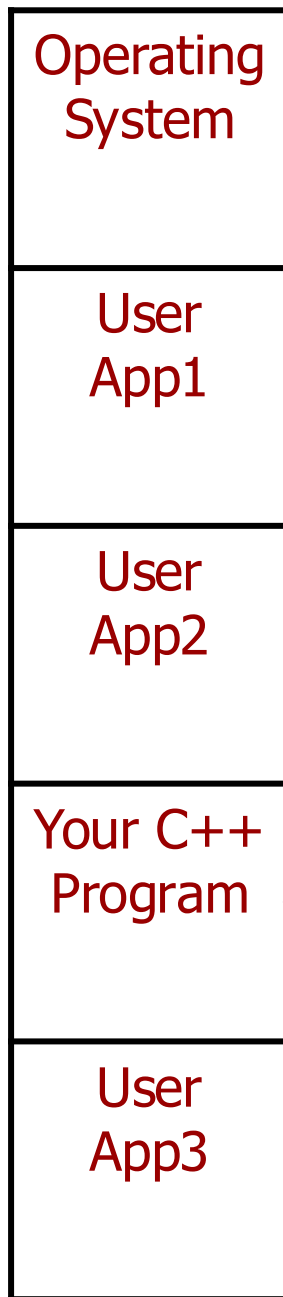


All of
main
memory



All of
main
memory



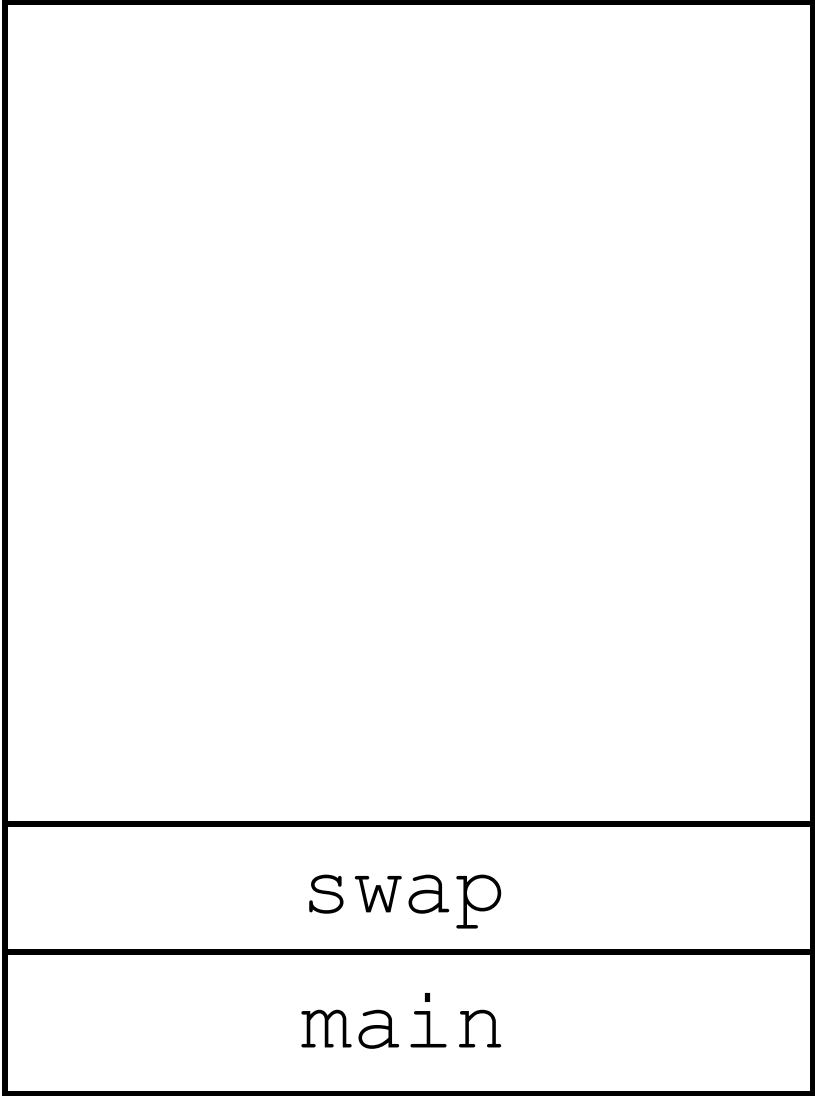


```
// Swap two int values.
void swap(int a, int b)
{
    int temp = a; // store a in temp
    a = b;         // put b into a
    b = temp;     // put temp a into b
}

int main()
{
    int x = 12;
    int y = 33;
    swap(x, y);
    cout << "x == " << x << "   y == " << y << endl; // ?
    return 0;
}
```



main





main





main

12
x

33
y



swap

12
a

33
b

main

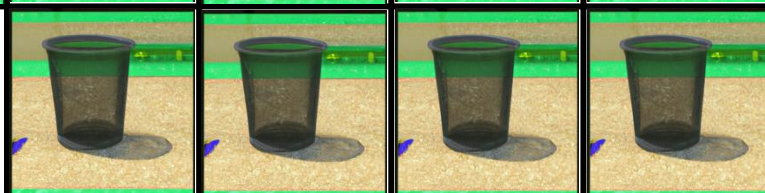
12
x

33
y



swap

12
temp



12
a

33
b

main

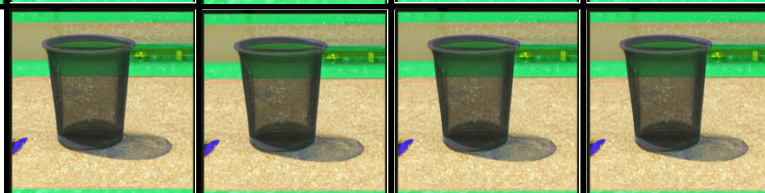
12
x

33
y



swap

12
temp



33
a

33
b

main

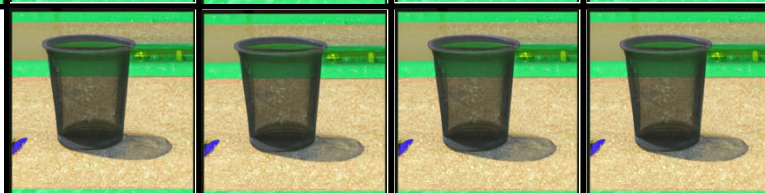
12
x

33
y



swap

12
temp



33
a

12
b

main

12
x

33
y



main

12
x

33
y

```
// Swap two int values.
```

```
 void swap(int* a, int* b)
```

```
{
```

```
     int temp = *a;    // store contents of a in temp
```

```
     *a = *b;         // put contents of b into a
```

```
     *b = temp;       // put temp a into b
```

```
}
```

```
int main()
```

```
{
```

```
    int x = 12;
```

```
    int y = 33;
```

```
    swap(&x, &y);    // pass by reference (the addresses of x and y)
```

```
    cout << "x == " << x << "   y == " << y << endl;
```

```
    return 0;
```

```
}
```





main

12
x

33
y



swap

0x45FA137D

b

0x45FA1379

a

main

0x45FA1379

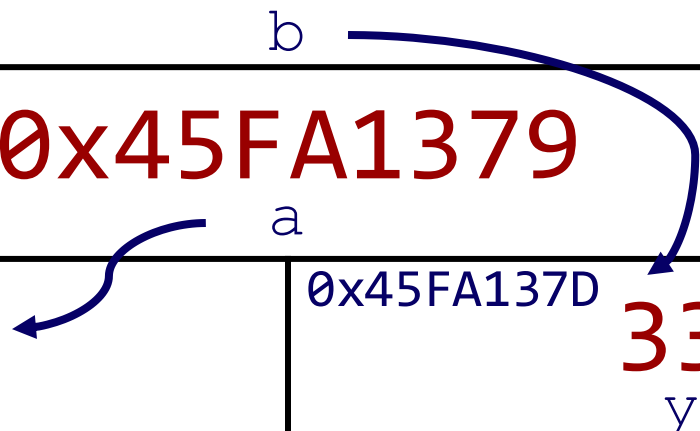
12

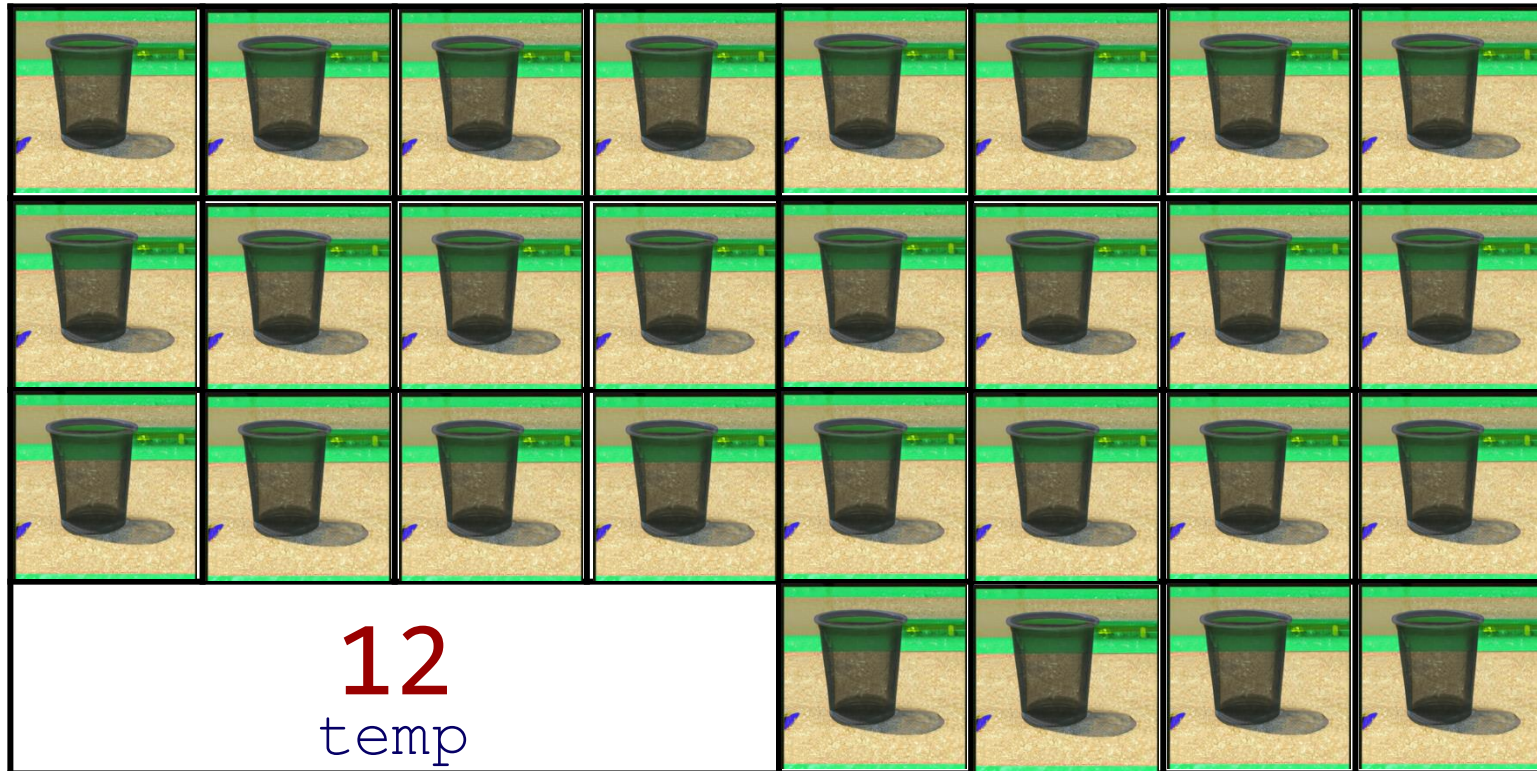
x

0x45FA137D

33

y





swap

0x45FA137D

b

0x45FA1379

a

main

0x45FA1379

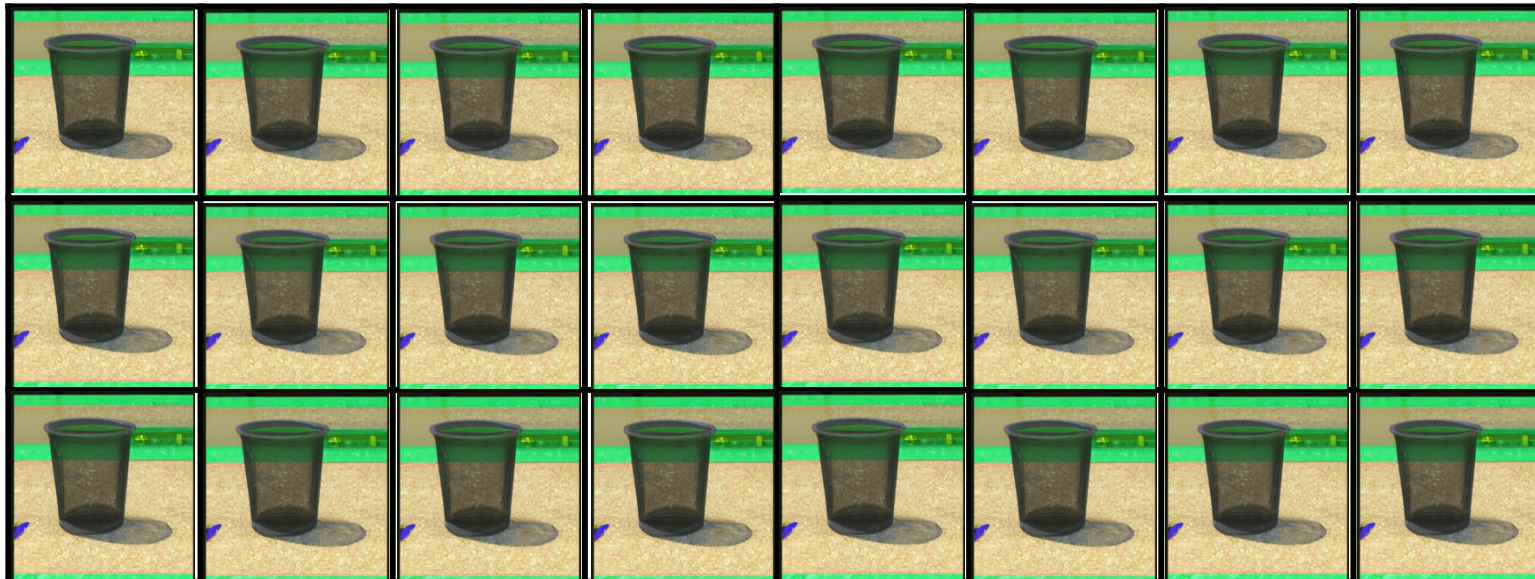
12

x

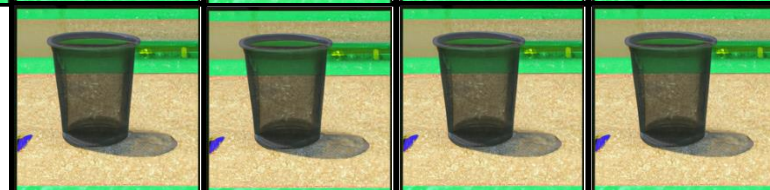
0x45FA137D

33

y



12
temp



swap

0x45FA137D

b

0x45FA1379

a

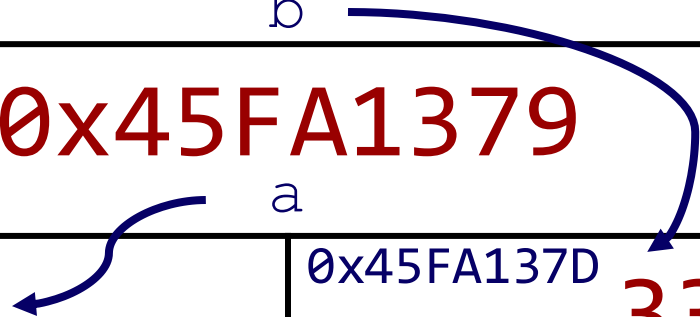
main

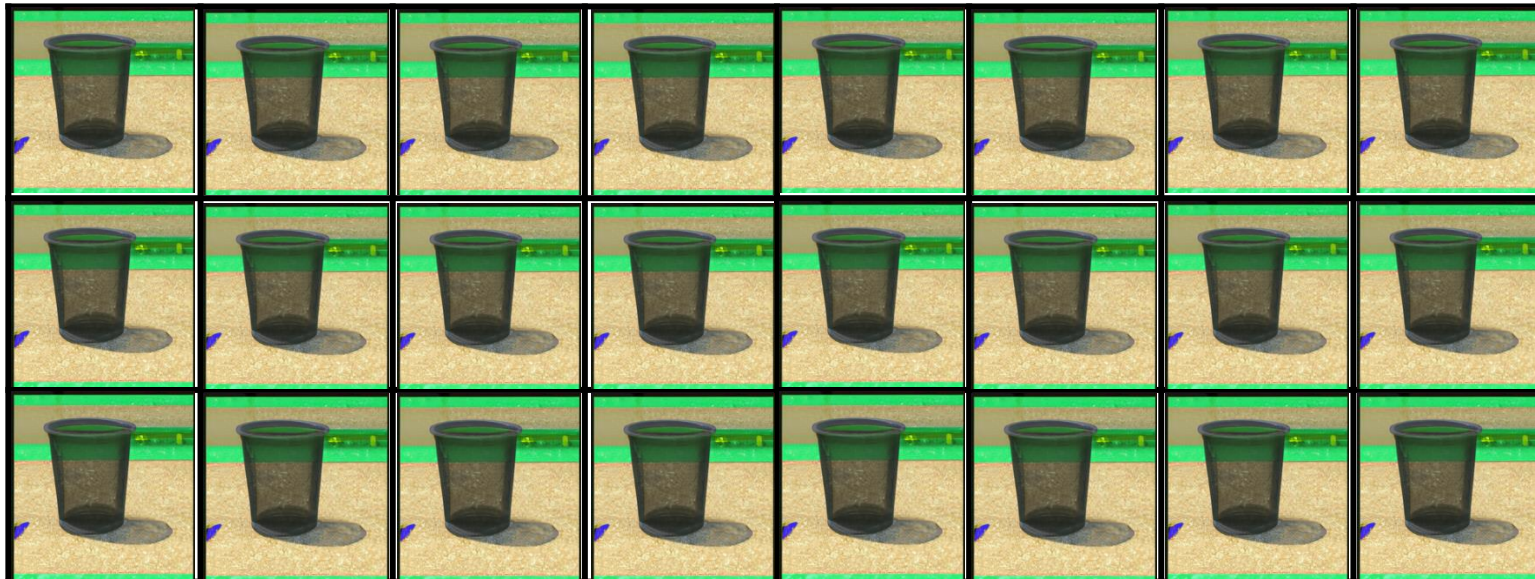
0x45FA1379

33
x

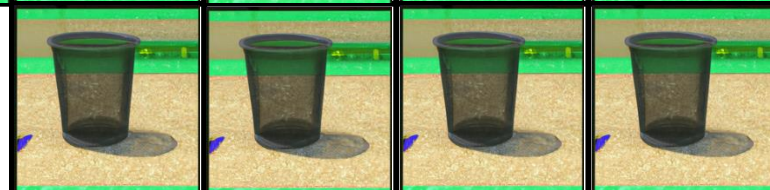
0x45FA137D

33
y





12
temp



swap

0x45FA137D

b

0x45FA1379

a

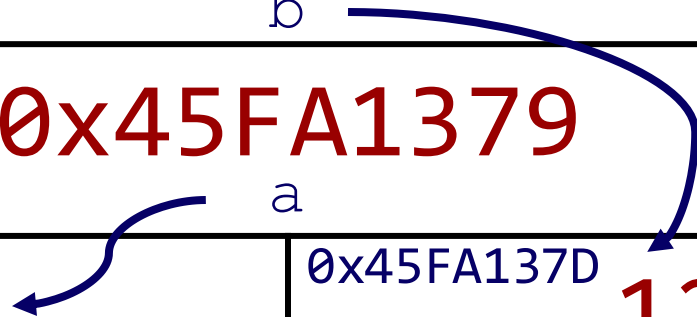
main

0x45FA1379

33
x

0x45FA137D

12
y





main

33
x

12
y

```
// Swap two int values.
```

```
 void swap(int* a, int* b)
```

```
{
```

```
     int temp = *a;    // store contents of a in temp
```

```
     *a = *b;         // put contents of b into a
```

```
     *b = temp;      // put temp a into b
```

```
}
```

```
int main()
```

```
{
```

```
    int x = 12;
```

```
    int y = 33;
```

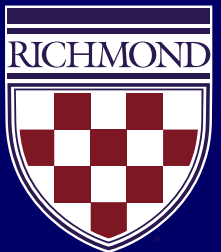
```
    swap(&x, &y);    // pass by reference (the addresses of x and y)
```

```
    cout << "x == " << x << "   y == " << y << endl;
```

```
    return 0;
```

```
}
```

Ask a question



Today

- ~~Memory~~

- ~~Pointers~~



- ~~Garbage Values~~

- ~~Memory Layout~~

- In-class exercise



Credits

- Malan CS50 
 - Computer memory image and yellow grid
 - Lecture materials
- Open-AI 
 - 3-D rendered garbage can image
- xkcd.com
 - Pointers comic: <https://xkcd.com/138/>
- Unsplash.com
 - Image of post office boxes