Data Structures

CSCI 2270

Algorithms, Dynamic Memory, Pointers

Sprint 3

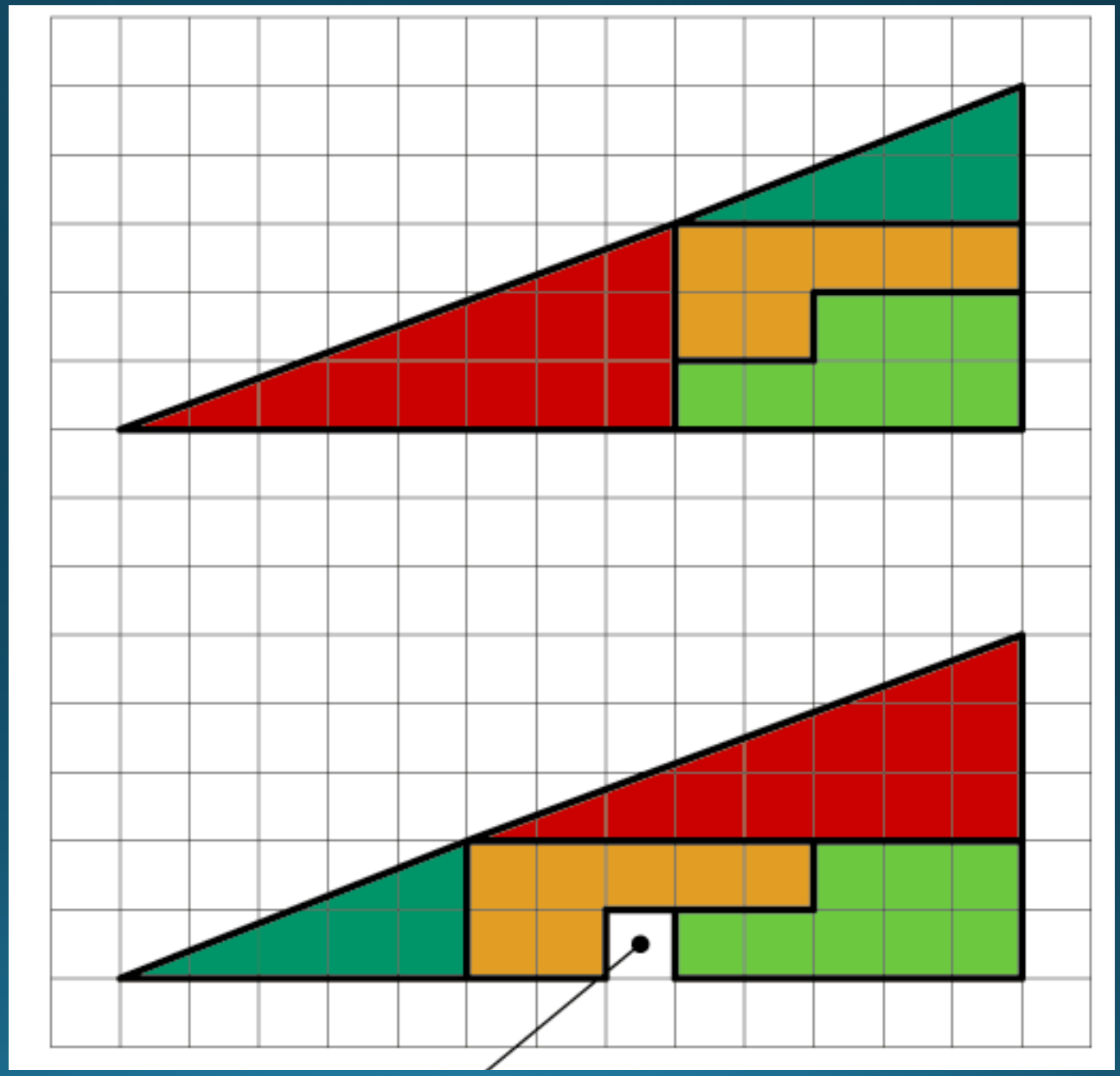
....

Ready

....

GO!

Howwww ??



Pointers

- Variables that store address
- *
- &
- type *var

Pointers

```
int myInt = 5;
cout<<myInt<<endl;

int *myPtr = myInt;
cout<<myPtr<<endl;

.....
```

Pointers

Address	Value	Variable
0xFFFF		
0xFFFE		
0xFFFD		ptrX
0xFFFC		
0xFFFB	0xFF	
0xFFFA	0x01	
.		
.		
.		
0xFF04	0xFF	X
0xFF03	0x0A	
0xFF02	0x01	
0xFF01	0x05	
0xFF00		

Struct

- Collection of data elements
- Composite object
- Give us more flexibility

Struct

```
struct WeatherData
{
        double temp;
        double humidity;
        double windVelocity;
}
```

Struct

```
 struct WeatherData
{
        WeatherData();
}
// create an instance
WeatherData dp1;
```

Array

- Data Structure
- Static Data
- Contiguous data in memory

Array

- What if we need more flexibility?
- What if we don’t want to waste space?
- What if we don’t know the size we need to deal with?

Array

- Effectively use dynamic pointers
  - int * myPtr = new int //pointing to int
  - int * myPtr = new int[3] //pointing to array of int

- Remember to free the memory allocated!!!
  - We are in the Heap

Array

Example:

command line arguments

```
int main(int argc, char *argv[])
{
        ....
}
```

Array

Example:

command line arguments

```
int main(int argc, char *argv[])
{
        // how do we access the argument?
}
```

Array

Example:

command line arguments

```
int main(int argc, char *argv[])
{
        // how do we convert to something
        // meaningful?
}
```

Lecture Quiz

Key: zippo