Quick Preliminaries

Get the example CODES

Linux terminal type:

```
git clone https://github.com/soam5515/CPPTalk2015.git
```

switch to the directory with all the code in it by typing:

```
cd CPPTalk2015
```

Compile things by typing:

```
make
```

Windows got to:

https://github.com/soam5515/CPPTalk2015

Press the download Zip button

```
(*(&a[0]+1))->GetTalk();
```

What are C and C++

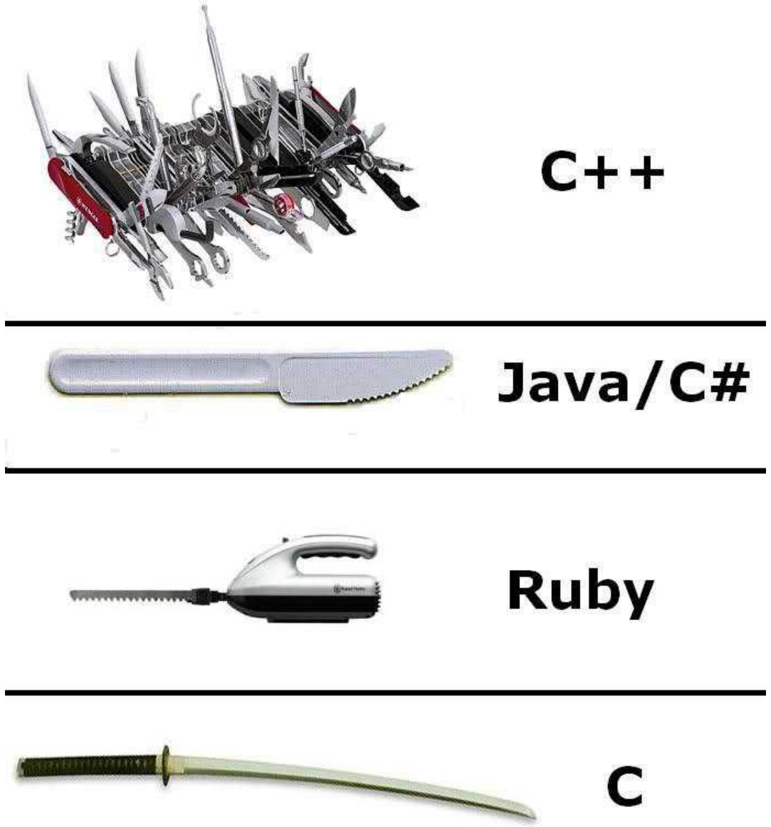
C is the predecessor to C++. It is a:

- Compiled
- Strongly typed
- Statically typed
- Lean mean fighting machine
- Everything is written in C including C

C++ is the object oriented upgrade to C.

- It contains the entirety of C
- It is object oriented (has classes and objects)
- It Implements everything

Both are heavily **context based** languages



Scope of this Presentation

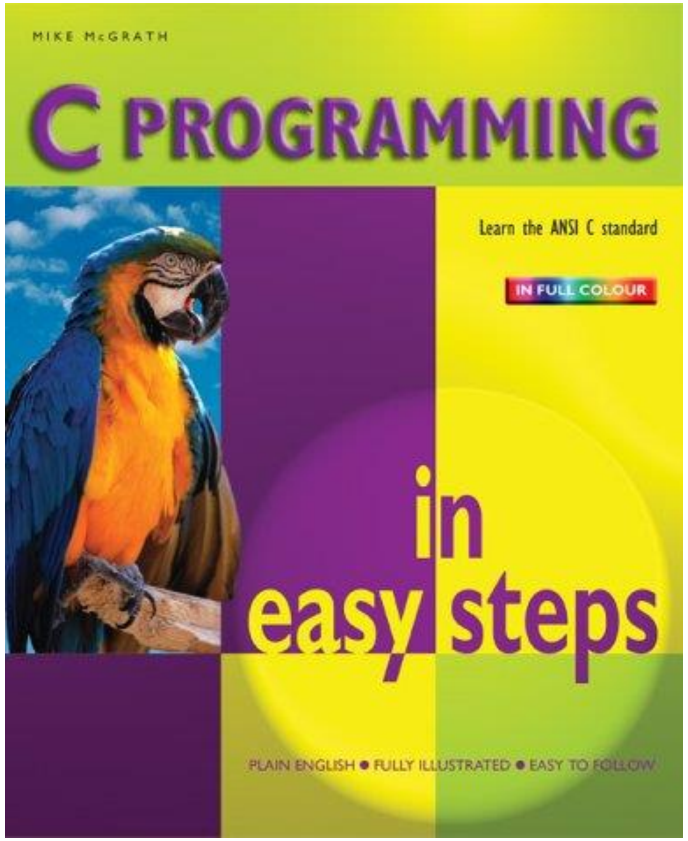
Start with some low level C concepts and refine our understanding

Memory addresses
Pointers
C arrays
Why C arrays are bad
Memory allocation
Tips and tricks

This week

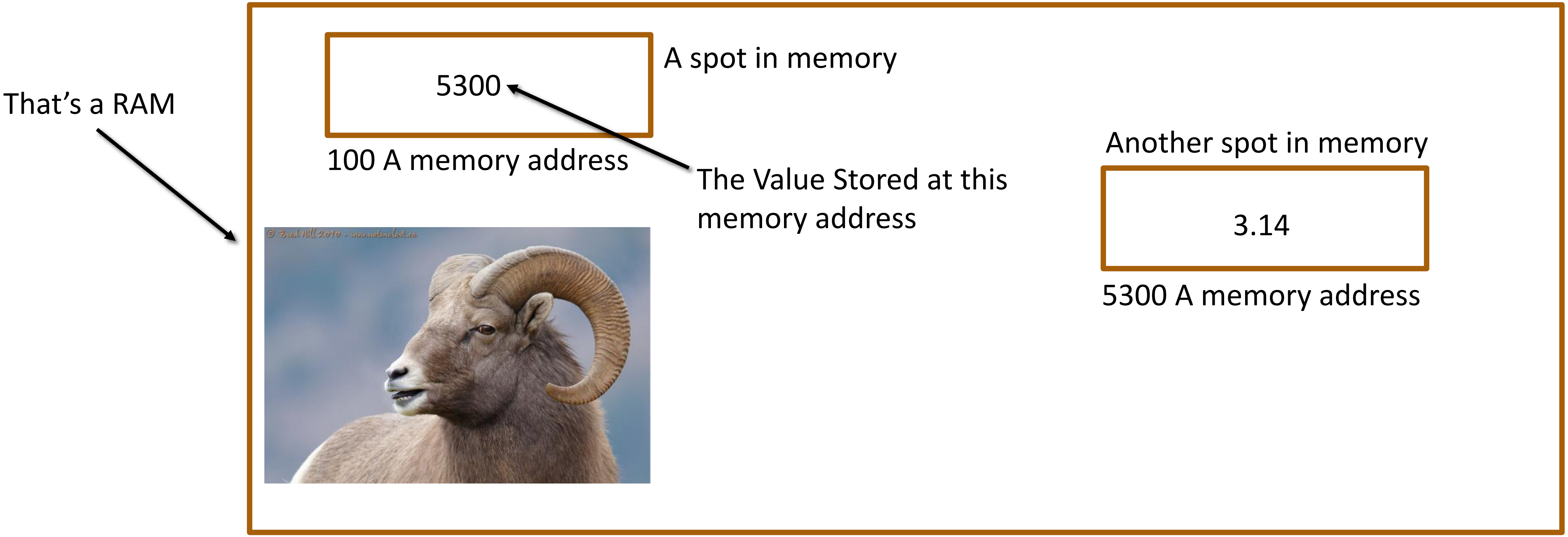
The Standard library
Vectors, Maps Galore
Iterators
C++11

Next week


MIKE McGRATH
C PROGRAMMING
Learn the ANSI C standard
IN FULL COLOUR
in easy steps
PLAIN ENGLISH ● FULLY ILLUSTRATED ● EASY TO FOLLOW

Parrot not included

The Most Beautiful Diagram in the World



That’s a RAM

5300

A spot in memory

100 A memory address

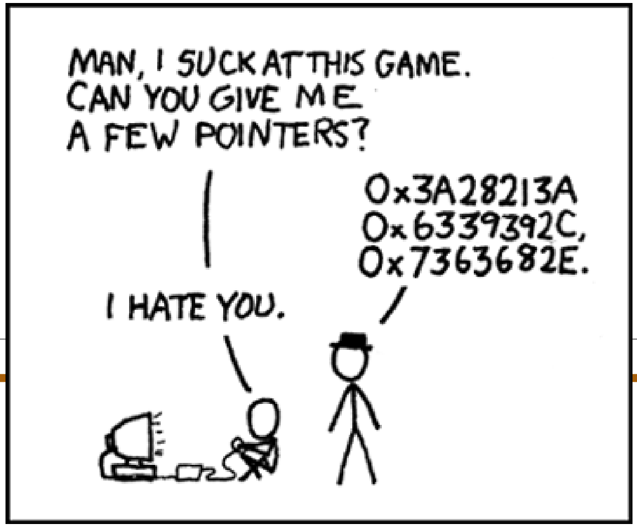
The Value Stored at this memory address

Another spot in memory

3.14

5300 A memory address

It’s a POINTER


MAN, I SUCK AT THIS GAME. CAN YOU GIVE ME A FEW POINTERS?
0x3A28213A 0x6339392C, 0x7363682E.
I HATE YOU.

That’s a RAM

5300

A spot in memory

100 A memory address

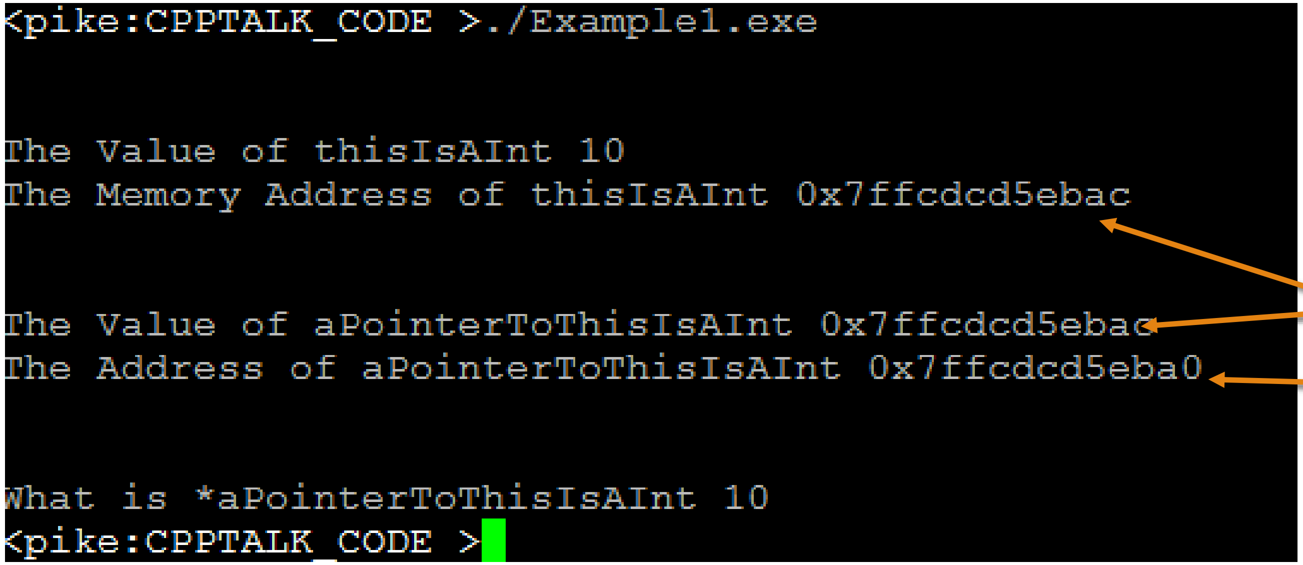
Another spot in memory

3.14

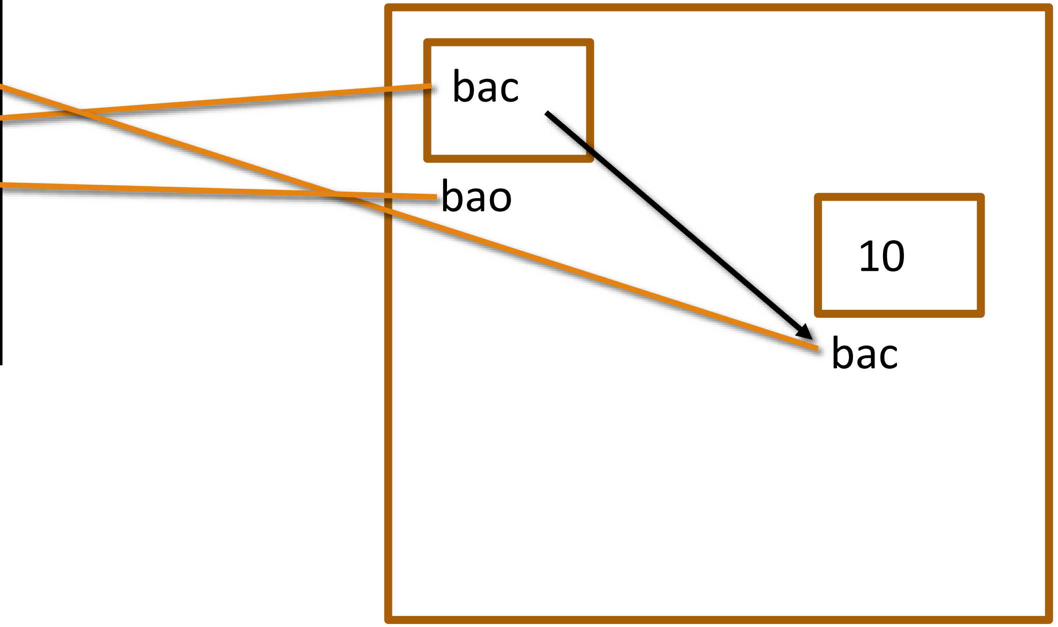
5300 A memory address

```
cout<<p;   → 5300
cout<<&p; → 100
cout<<*p;  → 3.14
```

Example 1



```
<pike:CPPTALK_CODE >./Example1.exe


The Value of thisIsAInt 10
The Memory Address of thisIsAInt 0x7ffcdcd5ebac


The Value of aPointerToThisIsAInt 0x7ffcdcd5ebac
The Address of aPointerToThisIsAInt 0x7ffcdcd5eba0


What is *aPointerToThisIsAInt 10
<pike:CPPTALK_CODE >
```



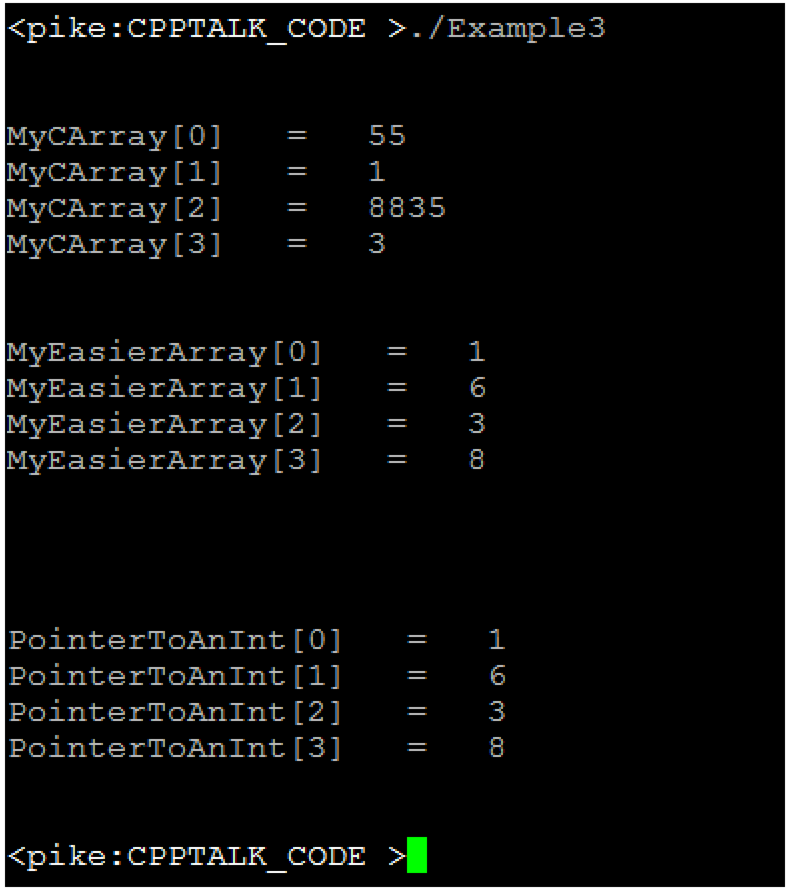
Example 3 (Example 2 has been cut)

In C Pointers and Arrays are *essentially* the same thing

Wat?

You see they are the same thing

```
int MyEasierArray[]={1,6,3,8};



int * PointerToAnInt =MyEasierArray;

for (int i=0;i<4;i++){
  cout<<"PointerToAnInt["<<i<<"]   =   "<<PointerToAnInt[i]<<endl;

}
cout<<endl<<endl;
```



```
<pike:CPPTALK_CODE >./Example3


MyCArray[0]    =   55
MyCArray[1]    =   1
MyCArray[2]    =   8835
MyCArray[3]    =   3


MyEasierArray[0]    =   1
MyEasierArray[1]    =   6
MyEasierArray[2]    =   3
MyEasierArray[3]    =   8




PointerToAnInt[0]    =   1
PointerToAnInt[1]    =   6
PointerToAnInt[2]    =   3
PointerToAnInt[3]    =   8


<pike:CPPTALK_CODE >
```

Exercise 1 C-Arrays

Open it

Compile it

- Linux type make
- Windows press the magic button in pocket C++

Run it

- Linux type: ./Exercise1.exe (I put the exe there to make it feel like windows)
- Windows press the other magic button

There are TWO Questions in the .cpp file. Feel Free to answer them and play around with the code

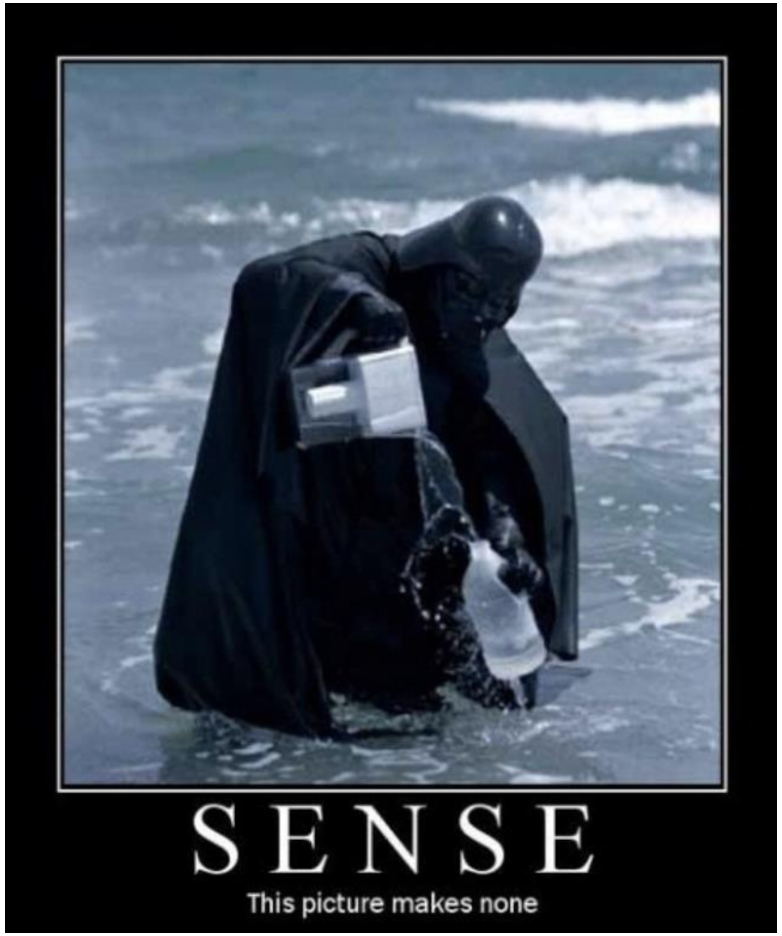
Smooth Jazz 5ish minutes

3[MyEasierArray] WAT?

This works because the [] notation just does pointer arithmetic.

- You can start at address MyEasierArray and move 3 spaces OR
- You can start at address 3 and move MyEasierArray spaces

The answer to question two is YES the array is defined by its starting address



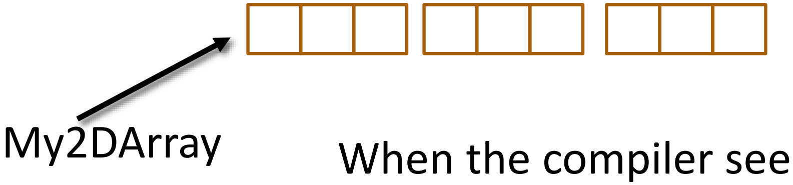
Exercise 2   2-D CArrays

There are 3 questions

Remember the trace of a matrix is the sum of its diagonal values

Smoother Jazz 5ish minutes

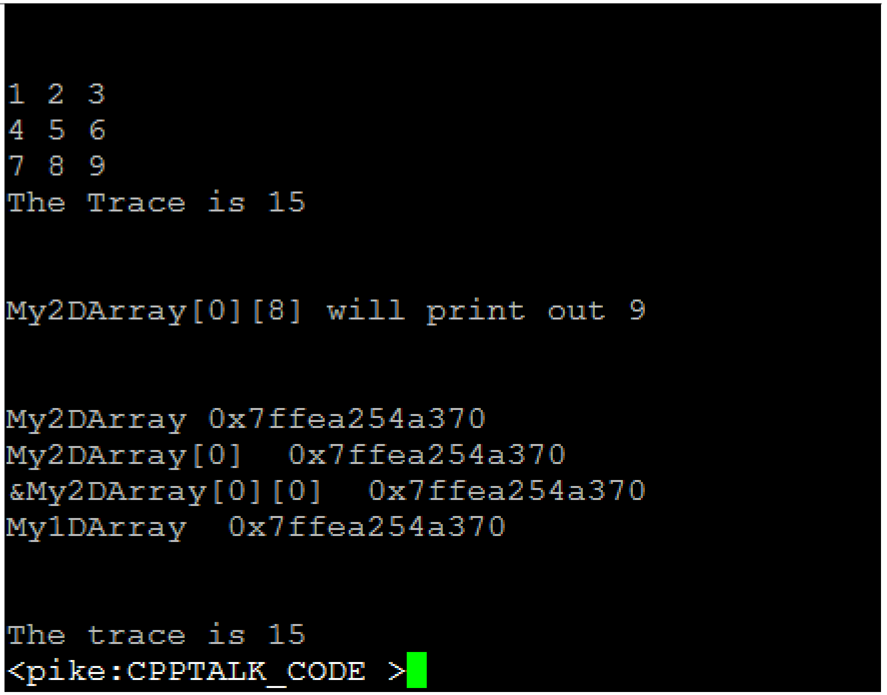
Exercise 2 Solutions



My2DArray

When the compiler sees My2DArray[0][8] it says 0 in the first index so it doesn't go anywhere than it sees 8 and moves to the end of the array

My1DArray[i*3 +j]
has the same effect



```
1 2 3
4 5 6
7 8 9
The Trace is 15


My2DArray[0][8] will print out 9


My2DArray 0x7ffea254a370
My2DArray[0]  0x7ffea254a370
&My2DArray[0][0]  0x7ffea254a370
My1DArray  0x7ffea254a370


The trace is 15
<pike:CPPTALK_CODE >
```

Exercise 3 It should seg fault

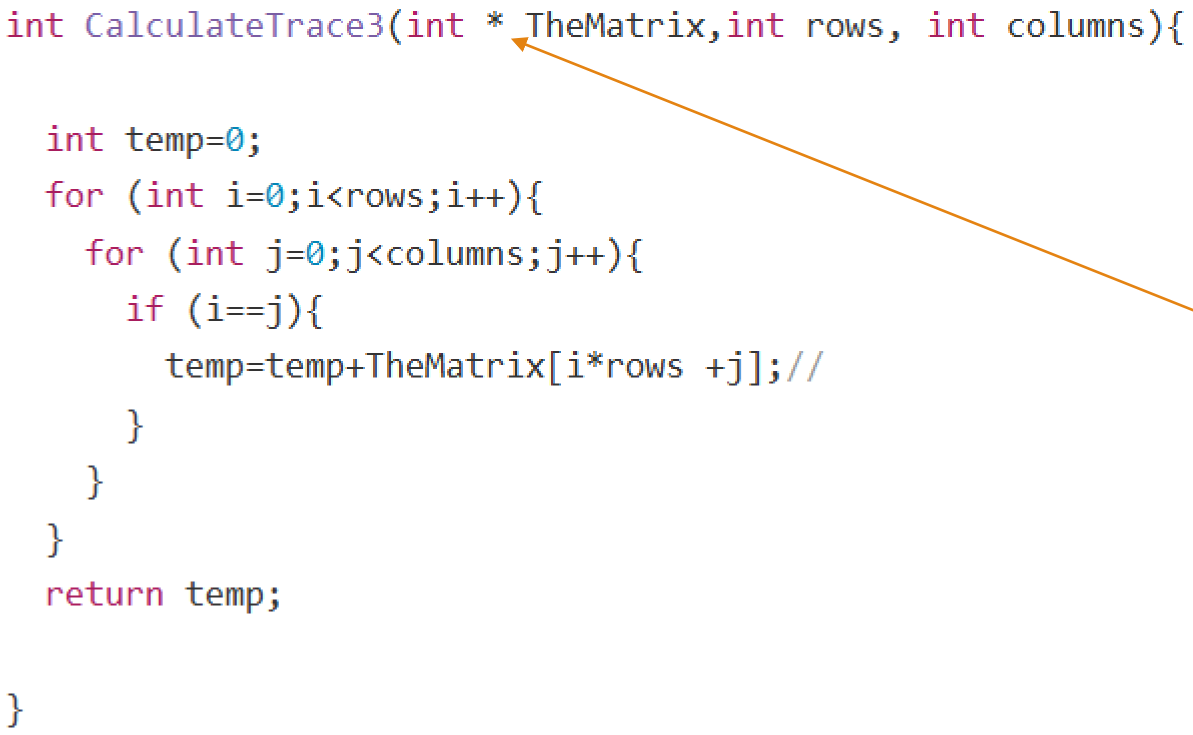
To get rid of the seg fault comment out the call to CalculateTrac2

- Think about why you this doesn’t work

Than answer the question at the bottom

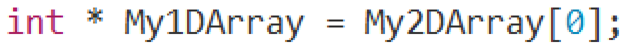
Smoothest Jazz   5ish Minutes

Exercise 3 Solutions



```
int CalculateTrace3(int * TheMatrix,int rows, int columns){

  int temp=0;
  for (int i=0;i<rows;i++){
    for (int j=0;j<columns;j++){
      if (i==j){
        temp=temp+TheMatrix[i*rows +j];//
      }
    }
  }
  return temp;

}
```

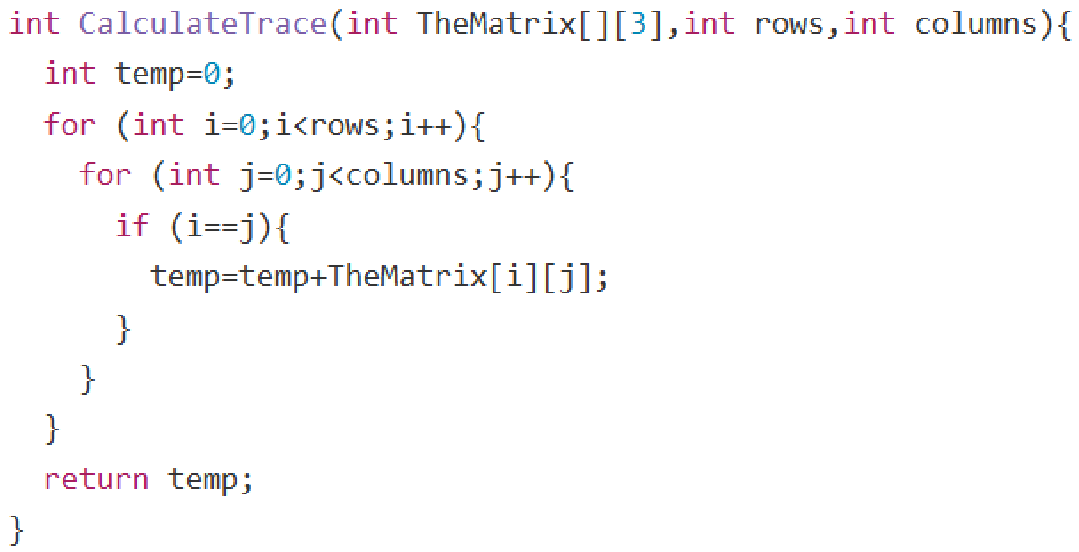
In Main



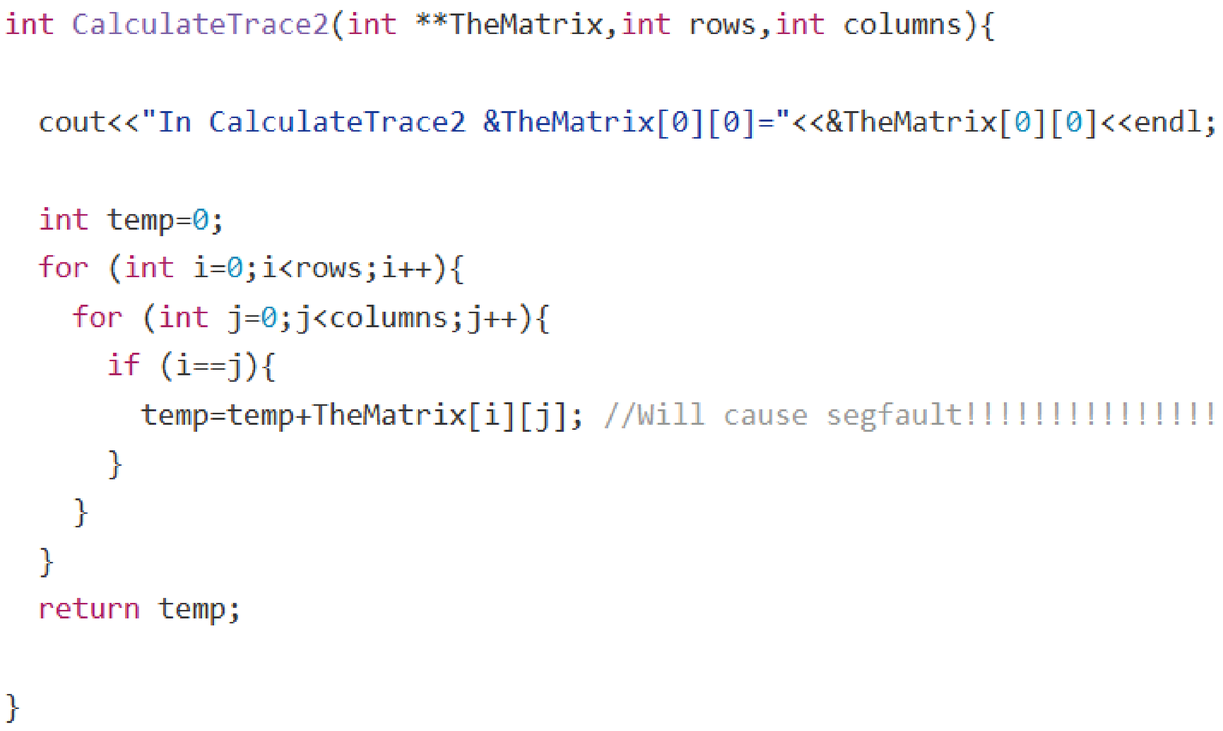
```
int * My1DArray = My2DArray[0];
```

All you need to do is set a int* pointer to array and pass that point to CalculateTrace3

More Exercise 3 and Recap



```
int CalculateTrace(int TheMatrix[][3],int rows,int columns){
  int temp=0;
  for (int i=0;i<rows;i++){
    for (int j=0;j<columns;j++){
      if (i==j){
        temp=temp+TheMatrix[i][j];
      }
    }
  }
  return temp;
}
```



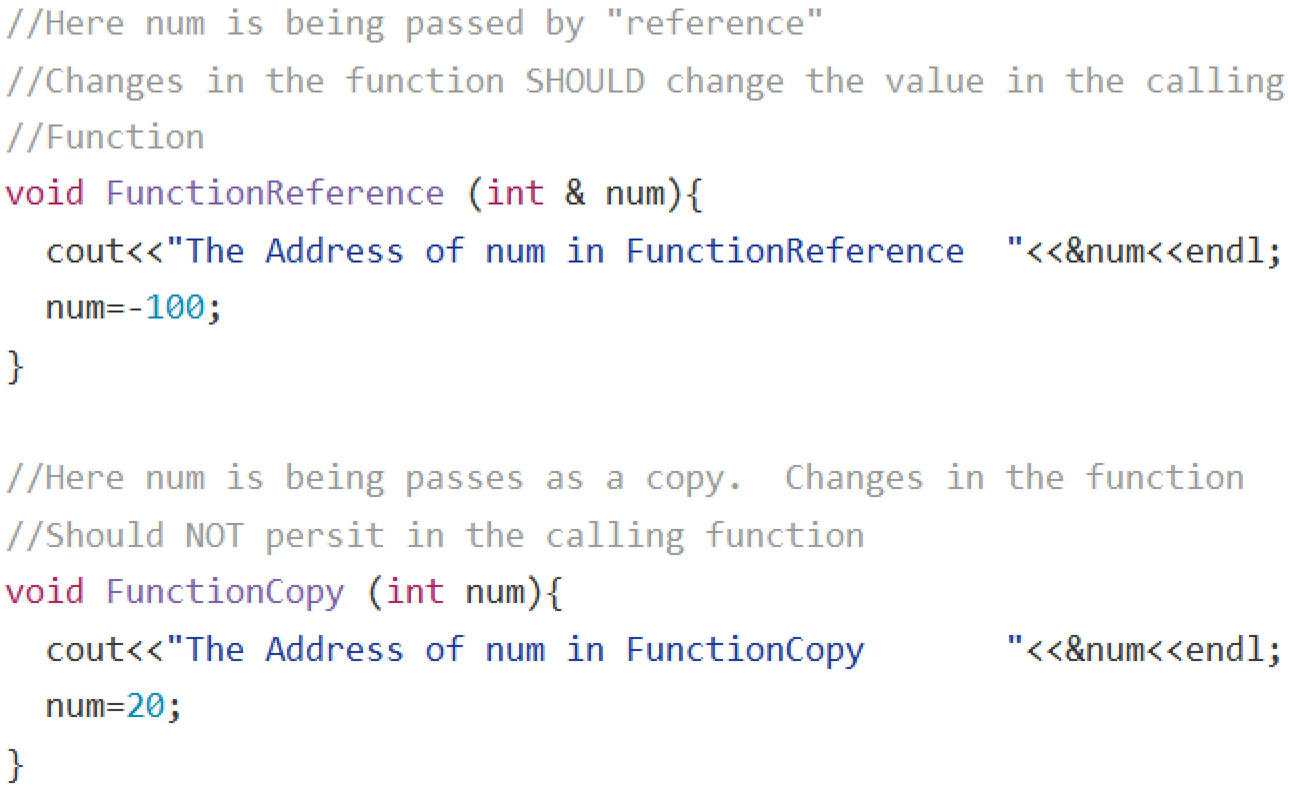
```
int CalculateTrace2(int **TheMatrix,int rows,int columns){

  cout<<"In CalculateTrace2 &TheMatrix[0][0]="<<&TheMatrix[0][0]<<endl;

  int temp=0;
  for (int i=0;i<rows;i++){
    for (int j=0;j<columns;j++){
      if (i==j){
        temp=temp+TheMatrix[i][j]; //Will cause segfault!!!!!!!!!!!!!!!
      }
    }
  }
  return temp;

}
```

Quick Tricks and Important Facts

The Stack vs the head

- There are TWO types of memory
- Go to StackVsHeap Code

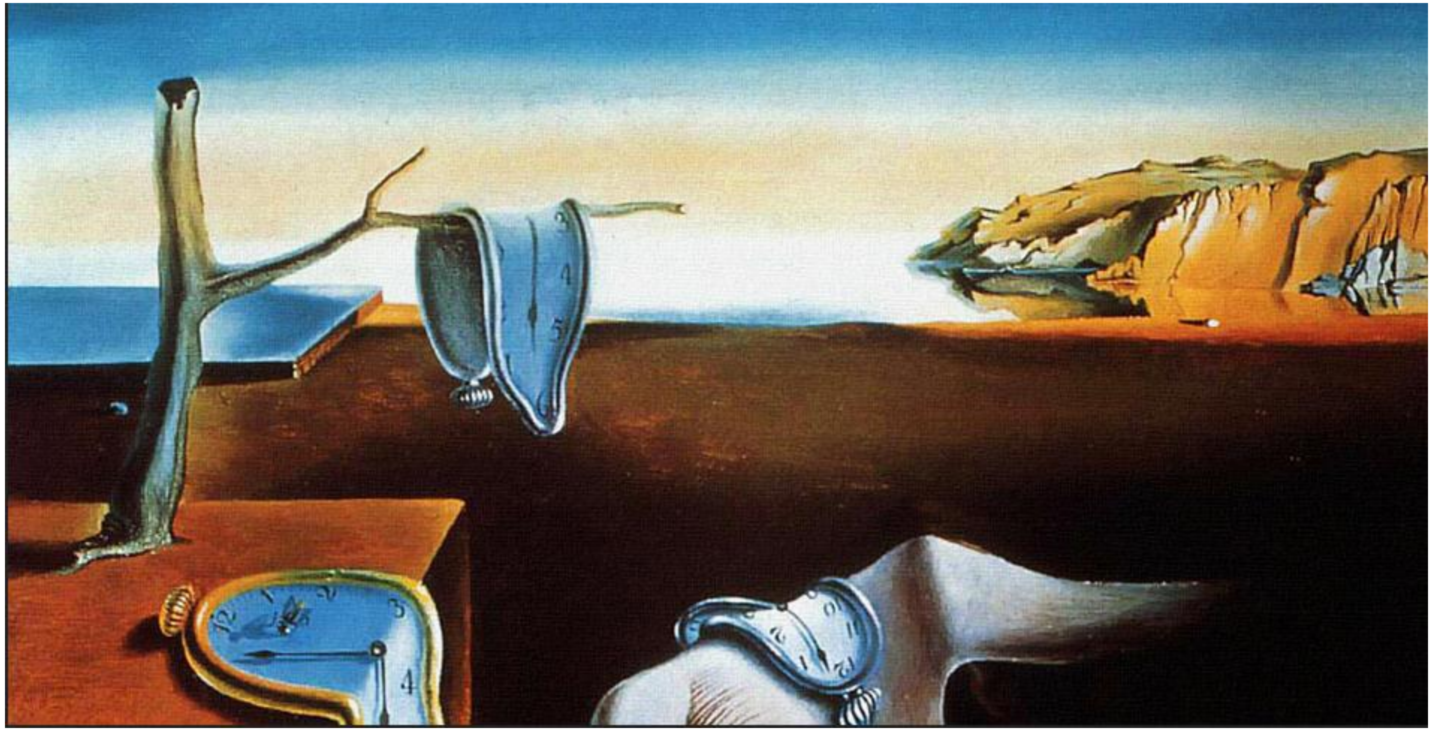
& can do so much more



```
//Here num is being passed by "reference"
//Changes in the function SHOULD change the value in the calling
//Function
void FunctionReference (int & num){
  cout<<"The Address of num in FunctionReference  "<<&num<<endl;
  num=-100;
}

//Here num is being passes as a copy.  Changes in the function
//Should NOT persit in the calling function
void FunctionCopy (int num){
  cout<<"The Address of num in FunctionCopy       "<<&num<<endl;
  num=20;
}
```

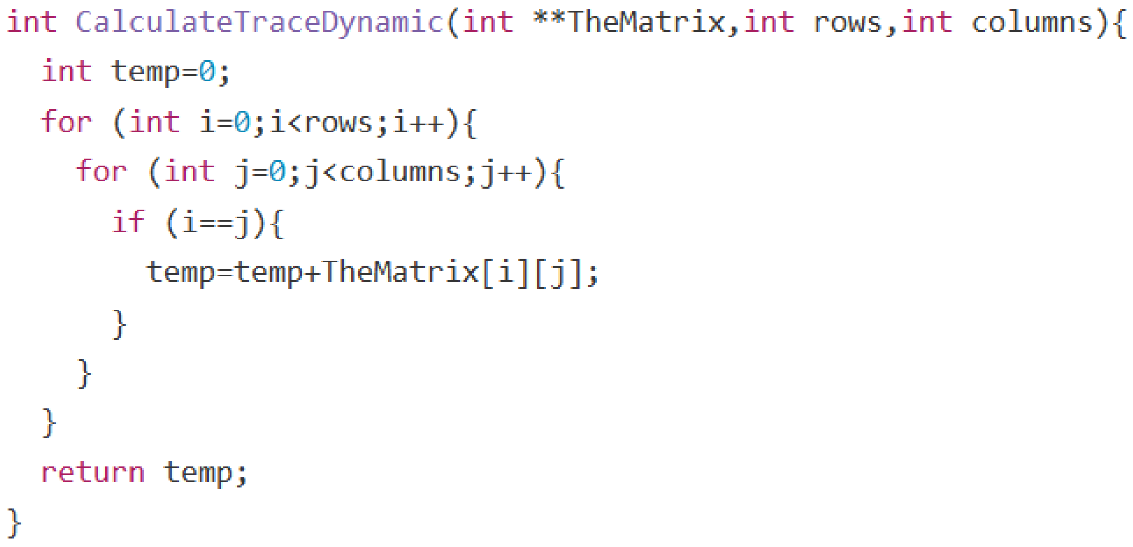
Exercise 4 Do I Still have time??!

Quick look at exercise 4 and think about the questions at the bottom



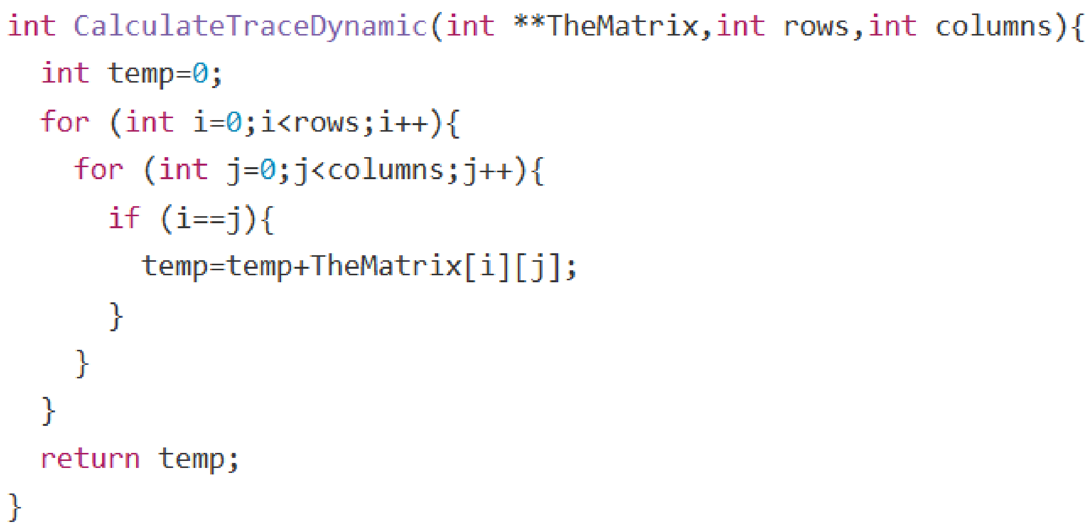
Exercise 4 Solutions

The Dynamically allocated Matrix that we made was NOT a continuous piece of memory necessarily. It is more like an array of arrays than the Matrix[3][3] declaration

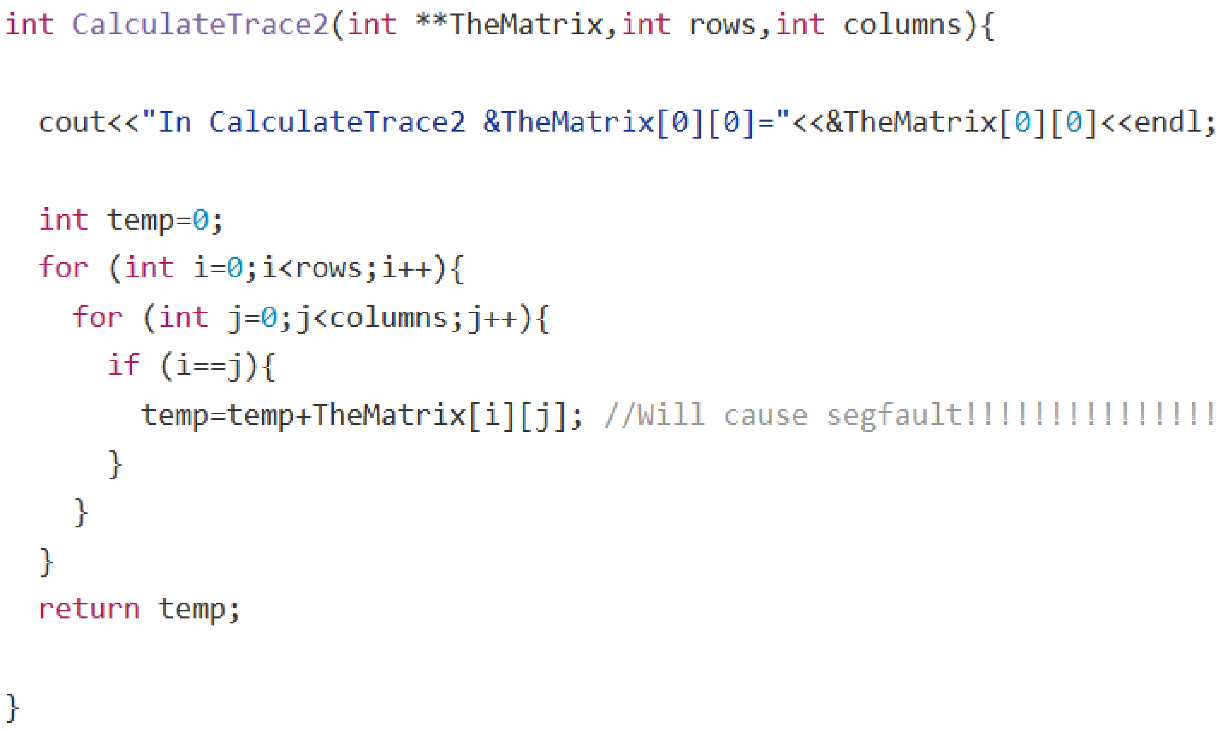


```
int CalculateTraceDynamic(int **TheMatrix,int rows,int columns){
  int temp=0;
  for (int i=0;i<rows;i++){
    for (int j=0;j<columns;j++){
      if (i==j){
        temp=temp+TheMatrix[i][j];
      }
    }
  }
  return temp;
}
```

Bonus slide



```
int CalculateTraceDynamic(int **TheMatrix,int rows,int columns){
  int temp=0;
  for (int i=0;i<rows;i++){
    for (int j=0;j<columns;j++){
      if (i==j){
        temp=temp+TheMatrix[i][j];
      }
    }
  }
  return temp;
}
```



```
int CalculateTrace2(int **TheMatrix,int rows,int columns){

  cout<<"In CalculateTrace2 &TheMatrix[0][0]="<<&TheMatrix[0][0]<<endl;

  int temp=0;
  for (int i=0;i<rows;i++){
    for (int j=0;j<columns;j++){
      if (i==j){
        temp=temp+TheMatrix[i][j]; //Will cause segfault!!!!!!!!!!!!!!!
      }
    }
  }
  return temp;

}
```

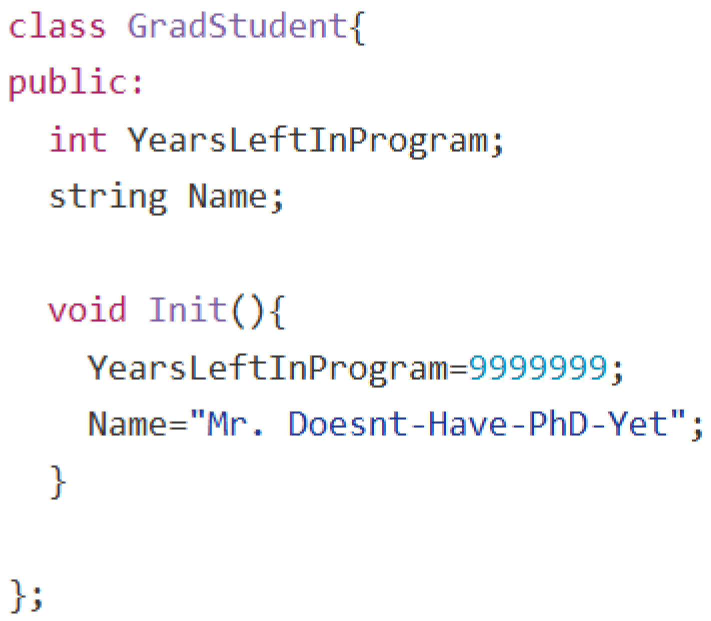
What I HAVE TIME!%!#^

What is object oriented programing?

- It’s using classes when write your programs

What the hell are classes?

- They are user defined types



```
class GradStudent{
public:
  int YearsLeftInProgram;
  string Name;

  void Init(){
    YearsLeftInProgram=9999999;
    Name="Mr. Doesnt-Have-PhD-Yet";
  }

};
```

Chris Sullivan to the Rescue

Exercise 5 has 3 intentional errors in it that Chris Sullivan will help you debug using

The GNU debugger GDB!#%!%@!#^@&@&