Discussion Section Week 2

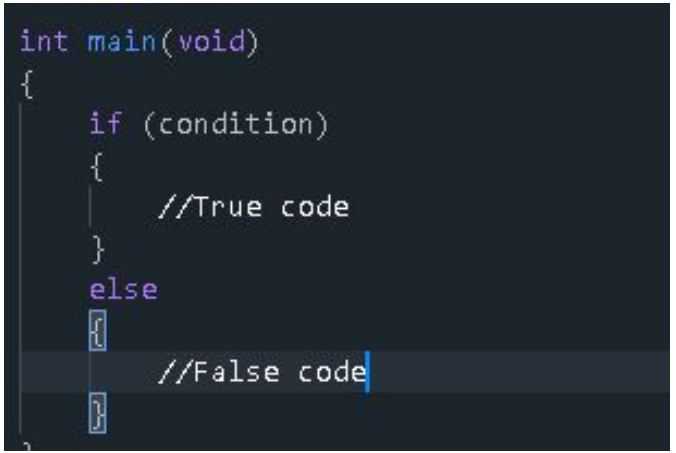
Conditionals, Loops, and Documentation

Context

- Conditionals and loops are essentially all coding is
- Mastering these skills is essential for you to become successful as a programmer
- Assignments….it’s a lot

Right into conditionals

- You know the basics



Beyond the Basics

- Different Conditional Formatting (Ternary Ifs, one liners)

```
void main(void)
{
    int test = 123456
    std::string out = ((test % 10) == 6) ? "Last digit is 6" : "Last digit is not 6";
}
```

result = (condition) ? true_branch : false_branch

```cpp
int main(void)
{
    int test = 123456;
    std::string out;

    if (test % 10 == 6)
    {
        out = "Last digit is 6";
    }
    else
    {
        out = "Last digit is not 6";
    }

    return 0;
}
```

```cpp
void main(void)
{
    int test = 123456
    std::string out = ((test % 10) == 6) ? "Last digit is 6" : "Last digit is not 6";
}
```

One Liners

```
int main(void)
{
    int temp = 49;
    if(temp % 2 == 0) std::cout << "Even" << std::endl;
    else std::cout << "Odd" << std::endl;
```

Exercise 1 (10 Min)

Write a program (named whatever you want) in c++ that does the following:

1) Takes in an integer and a string (Make sure the types are int and std::string)
2) Checks, using a ternary if statement, whether the string is a string representation of the int that was passed in
    a) If this condition is true, set a boolean flag to true, otherwise, set a boolean flag to false
    b) Hint: You will need to look up how to convert a string to an int
3) Checks, using “one liner” if else statments, whether that boolean flag is true or false
    a) If true, then output “Good”, otherwise output “Bad”

Example input: 12 12

Output: Good

How are conditionals (and loops too) executed?

No brackets? Assume the next line is iterated over/run on a condition

```
int main(void)
{
    int temp = 49;
    if(temp % 2 == 0) std::cout << "Even" << std::endl;
    else std::cout << "Odd" << std::endl;
```

```
int main(void)
{
    for(int i = 0; i < 5) ++i) std::cout << "Hello World" << std::endl;

    return 0;
}
```

Useful Things to Remember

- In C/C++, **everything** can be evaluated to either true or false
- Examples:
- if(1)
- if(myCharStr)
- if (!varName)
- while(str[i])
- etc

Loops

Again, you've seen the basics

While, For

While loops

Standard syntax:

while(condition) { body }

Iterating through arrays

```c
int main(void)
{
    char array[] = "Hello";

    int i = 0;
    while(array[i])
    {
        //perform operation on character
        i++;
    }

    return 0;
}
```

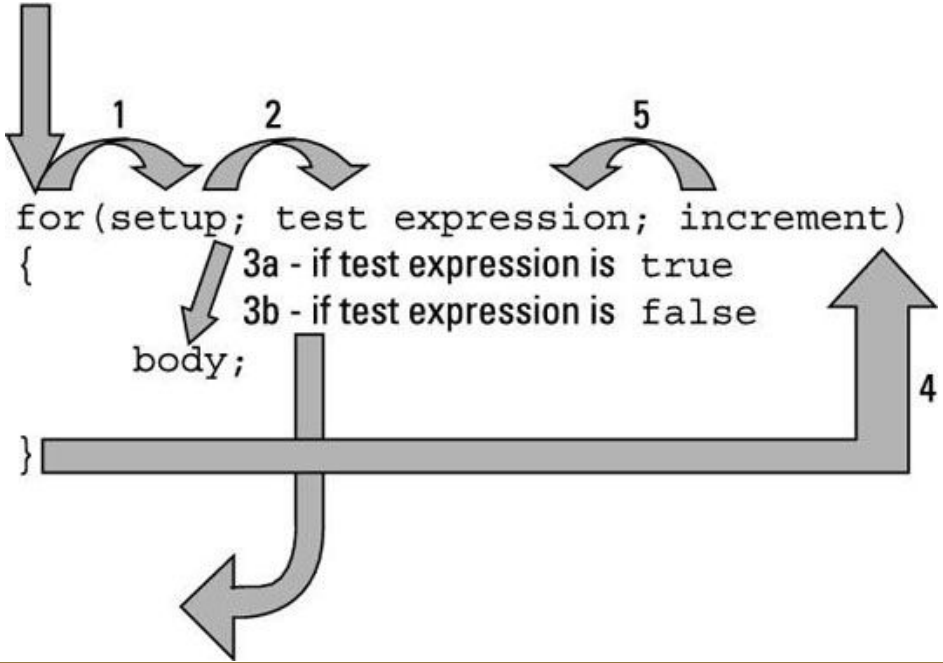
For Loops

- Range Based
- How to use them
- Increment

What's the difference in output?

```cpp
int main(void)
{
    for(int i = 0; i < 10; i++)
    {
        std::cout << "Hello" << std::endl;
    }

    for(int i = 0; i < 10; ++i)
    {
        std::cout << "Hello" << std::endl;
    }

    return 0;
}
```

For Loop Execution Order



Range Based For Loops

Similar to iterating through an array using a while loop

```
4  int main(void)
5  {
6      char str[] = "Hello";
7
8      for(auto ele: str) std::cout << ele << std::endl;
9
10     return 0;
11 }
```

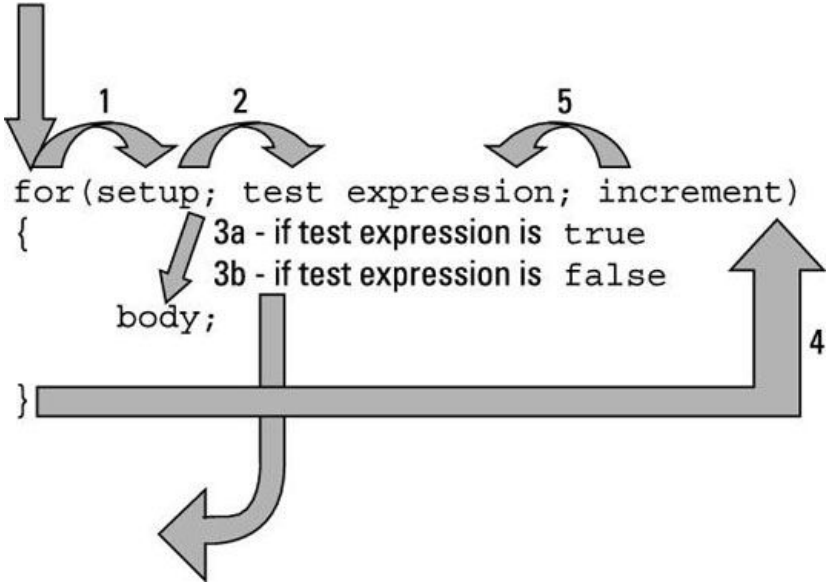
TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE

```
derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/TA$ g++ temp.cpp && ./a.out
H
e
l
l
o

derek@DESKTOP-3L8T6AU:/mnt/c/Users/Derek Jacobs/Desktop/CSC/TA$
```

Basic Syntax of Range Based For Loop

```
for (type varName : object)

{

        Body

}
```

More on Using For Loops




```
int main(void)
{
    char str[] = "aaaaaabbbbbbbccccc";
    std::string temp = "";

    for(int i = 0; str[i] != 'c'; ++i)
        temp += str[i];

    std::cout << temp << std::endl;

    return 0;
}
```

TERMINAL PROBLEMS OUTPUT DEBUG CONSOLE

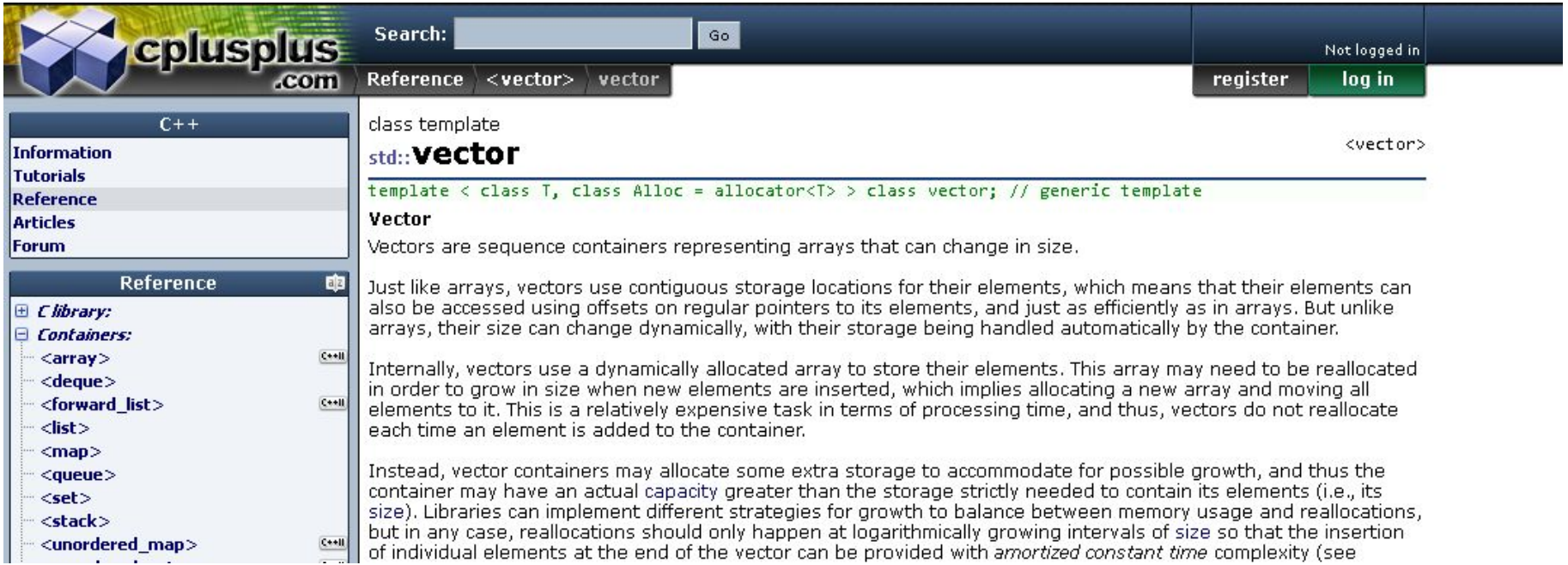
```
aaaaaabbbbbbb
```

```cpp
int main(void)
{
    int i = 0;
    std::string str = "Hello World";

    for(i; ; )
    {
        std::cout << i << std::endl;
        if(str[i] == 'l') i += 2;
        if (str[i++] == 'W') break;
    }

    std::cout << str[--i] << std::endl;
    return 0;
}
```

Looking up Documentation

https://en.cppreference.com/w/

http://www.cplusplus.com/reference/


cplusplus .com
Search:
Go
Not logged in
Reference
<vector>
vector
register
log in
C++
Information
Tutorials
Reference
Articles
Forum
Reference
C library:
Containers:
<array>
<deque>
<forward_list>
<list>
<map>
<queue>
<set>
<stack>
<unordered_map>
class template
std::vector
<vector>
template < class T, class Alloc = allocator<T> > class vector; // generic template
Vector
Vectors are sequence containers representing arrays that can change in size.
Just like arrays, vectors use contiguous storage locations for their elements, which means that their elements can also be accessed using offsets on regular pointers to its elements, and just as efficiently as in arrays. But unlike arrays, their size can change dynamically, with their storage being handled automatically by the container.
Internally, vectors use a dynamically allocated array to store their elements. This array may need to be reallocated in order to grow in size when new elements are inserted, which implies allocating a new array and moving all elements to it. This is a relatively expensive task in terms of processing time, and thus, vectors do not reallocate each time an element is added to the container.
Instead, vector containers may allocate some extra storage to accommodate for possible growth, and thus the container may have an actual capacity greater than the storage strictly needed to contain its elements (i.e., its size). Libraries can implement different strategies for growth to balance between memory usage and reallocations, but in any case, reallocations should only happen at logarithmically growing intervals of size so that the insertion of individual elements at the end of the vector can be provided with amortized constant time complexity (see

Final Exercise (15-20 Min)

Create a program that does the following:

1) Creates a vector of ints
    a) Hint: std::vector<type> varName;
    b) Hint: You'll need to include another library besides iostream
2) Using a "one-liner" for loop, append 10 random integers to your vector
    a) You'll need to call the push_back method
    b) You'll also need to call the rand() function
        i) This function is in the stdlib.h library
3) Find the maximum value in your vector using a range based for loop, output the max to your terminal

A note on rand()

It’s not actually random unless you seed it

To seed the random function

#include <time.h>

srand(time(NULL));

Now you’re values will be truly random

Submission

Please email your files to:

These are not graded for correction, only to see that you tried so you get credit for being here.