Session 6: Other Loop Types, Do Loops, and a Big Example
Session Outline

1. Quick Review
   1.1 For Loop
   1.2 Do While Loop
   1.3 Nested Loops
2. Times Table Example
While Loop

• By now you should be relatively familiar with the basic structure of the while loop:

```java
while (<this is true>) {
    <keep doing all this code>
    ...
}
```
Simple count controlled loop:

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

int main () {
    int x = 0;
    while (x < 30) {
        cout << "hello" << endl;
        x++;
    }
    return 0;
}
```
For loop – simple count controlled looping

#include <iostream>

using namespace std;

int main () {
    for (int x = 0; x < 30; x++) {
        cout << “hello” << endl;
    }
    return 0;
}
With for loops, one line does the work of 3!

While:
...
int x = 0;
...
while (x < 30) {
...
  x++;
Do While Loop

When you need to guarantee loop execution once before testing the condition, do while is there for you.

** As with many things in c++, typically you can restructure your code to work with whiles instead.**
int age;
cout << "enter age";
cin >> age;
while (age <= 25) {
    cout << "enter age";
    cin >> age;
}
Continuously Check if age is greater than 25 (While style)

```cpp
int age;

while (age <= 25) {
    cout << "enter age";    // **repeated
    cin >> age;            // **repeated
}
```
Continuously Check if age is greater than 25 (Do While style)

```cpp
int age;
do {
    cout << "enter age";
    cin >> age;
} while (age <= 25);

// slightly easier to read code, fewer lines!
```
Nesting Loops

As with if statements, we can nest loops:

```cpp
for (int x = 0; x < 30; x ++) {
    for (int y = 0; y < 30; y++) {
        cout << "*" << endl;
    }
}

*Good for creating tabular data
  • Think columns and rows. Inner = column. Outer = row.
Big example:

- Your little cousin is in the 4th grade and is learning multiplication:

  “This is impossible. No human can learn these. I bet a computer can't even do these!”

- You know some CS, you are ready to prove him wrong.

  “I will learn up to 4x multiplication tables if you can show me that computers can do these.”
Goal:

Write to the screen:

1 \times 1 = 1
1 \times 2 = 2
\ldots
1 \times 10 = 10
2 \times 1 = 2
\ldots
10 \times 10 = 100
Strategy:

There will be a number of different ways to approach this, some require a lot of code and some require just a bit.

Think of the algorithm and the techniques which can implement it, and start working through line by line.

You can always write “dumb code” first and then optimize later, it's just more time consuming.
Okay but...

“That took a lot of code! How can I learn it when it takes a computer that much work to learn?”

“I'll learn up to 7x tables if you can make a computer print it in under 15 lines of code!”

Using what you know about code refactoring, see if you can meet his crazy demand!
But wait...

“It just goes to the screen, and then it's gone?”
“I'll learn up to 10x tables if you can make it save the tables to a file!”
Remember our cipher example

• Can we make it work for strings instead of chars?
Simple Cipher Program

Write a program that allows a user to enter a string of lowercase letters and a encryption number.

Shift every character in the string ahead by the key number.

If a letter exceeds 'z', roll it over to 'a' and keep advancing.

Print the encrypted string.