**Objective**

Explore the basics of C++ programming. Create simple programs, which define variables, modify their values, read user input, and display results to the screen, as well as manipulate flow control.

You should only have 1 solution file and 1 project file for this lab.

Each program within each section ( 1-1,1-2, etc ) should be in its own function.

Do not just have all your code in main().

It is fine to have just one .cpp file ( the one that has your main() )

For this first lab do not use any classes.

LAB SUBMISSION: Delete all debug folders (usually one for solution and one for project) as well as .vs folder (if it is not hidden). Add the rest of the files to a zip file and submit the zip to FSO. Ensure that your submission contains all source code files (.cpp and .h)!

**Programs**

**Program 1-1**

Ask the user for their initials and save them into a char array. Then, ask the user for their age, and save it in an integer. Make sure that your age input is validated so that only numeric data will be accepted. HINT: Putting this validation into a function (aka method) will be helpful for other numeric inputs in this lab. Multiply the age by 365 to get the days that they’ve been alive (we don’t care about leap years, they’re worthless). Finally, display everything back to the user.

Example Output

Please enter your initials: TJ

Please enter your age: 999

TJ, did you know you’re at least 364635 days old?

**Program 1-2**

Make an array of integers of size 5. Then, ask the user to fill out each slot of the array with a number. Make sure to do input validation for each round of input. Once done, display the inputs back to the user. Again, make sure the user’s numeric input is validated. Be sure to use a loop to avoid repeated code.

Example Output

Please enter a number: 8

Please enter a number: 6

Please enter a number: 7

Please enter a number: 5

Please enter a number: 3

You entered: 86753

**Program 1-3**

Ask the user for 3 numbers and then use them to compute 2 separate but similar-looking equations. Here are the 2 equations you will compute:

    a + 1 \* b + 2 - c

    (a + 1) \* (b + 2) - c

Example Output

Please enter 3 numbers.

Enter value for a: 10

Enter value for b: 15

Enter value for c: 7

a + 1 \* b + 2 - c = 20

(a + 1) \* (b + 2) - c = 180

**Program 1-4**

Print out a table of ranges for each of the system-provided integral variable types (signed and unsigned). To help you get the ranges of each integral variable type (instead of just copying them from a book or the internet), use constants defined in C++. HINT: Typing out one of these constant definitions then right click -> ‘Go to Definition’ will show you the list of definitions.

SHRT\_MAX – gives you the minimum value of a short integer (16-bits)

INT\_MIN – gives you the minimum value of an integer (32-bits)

etc.

The point of this program is to introduce you to the fact that different variable types consume different amounts of memory. The numbers themselves may seem arbitrary now, but when you learn how much each variable type consumes, you will understand why those numbers are what they are.

Remember, unsigned numbers cannot be negative so all unsigned minimums are implied to be 0.

Example Output

Table of integral variable type ranges in C++:

type                       range

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

ushort                   0 to 65535

short                    -32768 to 32767

int                      -2147483648 to 2147483647

uint                     0 to 4294967295

longlong                 -9223372036854775808 to 9223372036854775807

ulonglong                0 to 18446744073709551615

**Program 1-5**

Create a char array of size 32. Ask the user for their full name, storing it in the array.  Make sure the input accepts strings that may include a space (HINT: cin.getline). Finally, display it back to them with some kind of interesting fact or something.

Example Output

Please enter your name: Guy McDude

Hello there Guy McDude, you’re interesting and full of facts!

**Program 1-6**

Write a program that first checks the user’s age before continuing. If the user is not old enough to run your program (say 16), then exit gracefully; otherwise, let them be scarred for life! Don’t get too crazy. Be sure to make proper use of if/else if/else chains here to reduce the amount of logical checks needed.

Example Output

How old are you? 13

I’m sorry, but you’re too young to play this game. Bye!

How old are you? 25

You’re a quarter of a century old!  Your life is slowly slipping away.

**Program 1-7**

Create a program to ask for and verify an even number. Check to see if the number they entered is in fact even, meaning the number is evenly divisible by 2 (i.e. no remainder). Still practice proper use of if/else code flow.

Example Output

Please enter an even number: 20

Nice!

Please enter an even number: 21

Boo!

**Program 1-8**

Create a program to ask for a divisor, then randomly generate 3 numbers and check each one to see if it is evenly divisible by the divisor. Use a loop to avoid repeating code. HINT: srand() should be called only one time in the entire program to allow for proper random number generation.

Example Output

Please enter a divisor: 7

130947868       - no

2083942967      - no

901599951       - yes!

**Program 1-9**

Create a program to ask the user which color Popsicle™ they would like. Use a switch statement, and possibly an enum, to process their input. Display an appropriate message for the desired color.

Example Output

What color Popsicle do you want from the freezer?

1)  Red

2)  Green

3)  Blue

13) Orange

> 1

Ah! Red cherry, good choice!

What color Popsicle do you want from the freezer?

1)  Red

2)  Green

3)  Blue

13) Orange

> 13

Uh... I don't know how long that's been in there...

**Program 1-10**

Create a program to simulate level difficulty. Each level of difficulty of your game will have a different number of enemies, though that number is random. Ask the user which level of difficulty they would like to play. Use a switch to process their choice. Generate a random number of enemies for your game, but generate a number in a higher range for harder levels of gameplay.

Example Output

Difficulty Levels

1)    Easy

2)    Medium

3)    Hard

What’ll it be? 1

You’ll have to fight 6 enemies!

Difficulty Levels

1)    Easy

2)    Medium

3)    Hard

What’ll it be? 3

You’ll have to fight 27 enemies!