EDUCATIONAL OUTREACH
FLORIDA POLYTECHNIC
UNIVERSITY

FLORIDA POLYTECHNIC

INTRODUCTION TO PROGRAMMING

Simple Math

Overview:

This lesson will teach you how to program a computer to do math calculations. By the end of the lesson, you will be able to do simple math equations such as add, subtract, multiply, and divide. This lesson is a great introduction to students who have not begun algebra. This lesson also serves as an algebra review working with variables and the order of operations they are involved in.

Learning Outcomes

-Student will complete code for simple math instructions in C

-Student will apply the completed code to solve mathematical equations

-Students will recognize data types, differentiate between them, and apply the data types in code

Materials

- Computer with internet connection
- Website for the online compiler https://replit.com/languages/c
- Vocabulary List (optional)

The Activity

Part 1

Begin with reviewing a brief explanation of what data types are and what they do. Data types were introduced in lesson 1 in the form of int and char. These are called data types because they define what traits a variable has. For example, an int makes a variable hold a number value, while a char can only hold characters. This is important to know in the future because each data type has their own traits that can change how a program functions. This lesson will need two data types: int which you should be familiar with and double. The double data type does not do what you may be thinking it does, it does not double the value within it, rather it holds a larger maximum number than int does and it also can contain decimal points, unlike int.

For future reference here is a short list of data types, what they do, how to reference them and how to declare them.

Data Type	What they do	Reference	Declaration
Int	Contains an integer that can be either positive or negative. Cannot have decimal points	%d	int x = 0;
Char	Contains characters that are either the alphabet or punctuation.	%c	char x = ‘A’;
Long	Has the same properties as an integer but has a vastly larger maximum value.	%li	long x = 0;
Double	Referred to as a floating point, which makes its max value much greater than an int, can also contain decimal points.	%f	double x = 0;

At this point, you should know a little more about data types and how they affect variables. With this information, the next step is learning how to make our computer do math for us. It is important to know that just like in math class, there are rules to math on a computer. Just like you, a computer follows the rules of PEMDAS where it will do the commands you input in **p**arentheses first, then **m**ultiplication and **d**ivision, and finally **a**ddition and **s**ubtraction. We will work with **e**xponents in this lesson.

Unlike you and I, a computer cannot look at a complex equation like $x = \frac{(9*10)+2}{12-8}$. We can easily read it and take the steps to complete the problem with little issue. But unlike us, a computer is methodical, meaning it reads everything from left to right. This is what the computer will see $x = (9 * 10) + {}^{2}/_{12} - 8$. As you might notice it is very different from what we see. This is because we need to be specific when doing arithmetic like this. To make the equation result in the same answer we will put parts of it in parentheses. For example, $x = ((9 * 10) + 2)/(12 - 8)$. This is an equation that the computer can read, and it will give us the same answer. It is also important that we place our sum or the answer that we want on the left side of the equal sign. This is because in computing and programming making a variable hold a value looks like this, Variable name = value you want to store.

Part 2

With a better idea of how a computer does math, try to program an equation. Start with simple addition.

Addition in *C* looks like this:

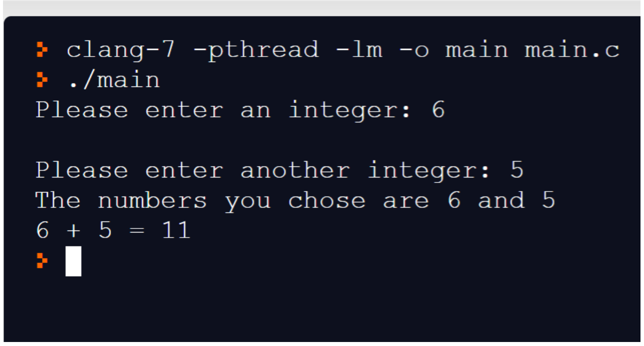
Sum = num1 + num2;

This is very simple, using what was learned in the lesson 1 we can make the computer tell us what it did using the printf function like so.

printf("%d + %d = %d\n", num1, num2, sum);

Test what you learned in this lesson so far to declare 3 int variables (num1, num2, and sum) and apply what you learned in the last lesson with the printf and scanf functions to prompt a user to enter 2 integers and add them together and show the output. If you get stuck you can ask your teacher, look back at the last guide as a reference, or you can look online for some tips and tricks to complete this code.

Your screen when you input the code should look like this:

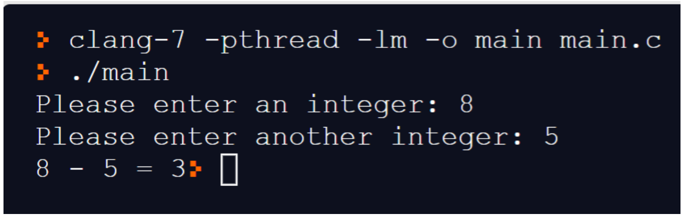


```
› clang-7 -pthread -lm -o main main.c
› ./main
Please enter an integer: 6

Please enter another integer: 5
The numbers you chose are 6 and 5
6 + 5 = 11
› █
```

Once completed, try what you just learned with subtracting two variables to get an answer. To get from what you just did to show subtraction is very easy. You will only need to change a few things in your code.

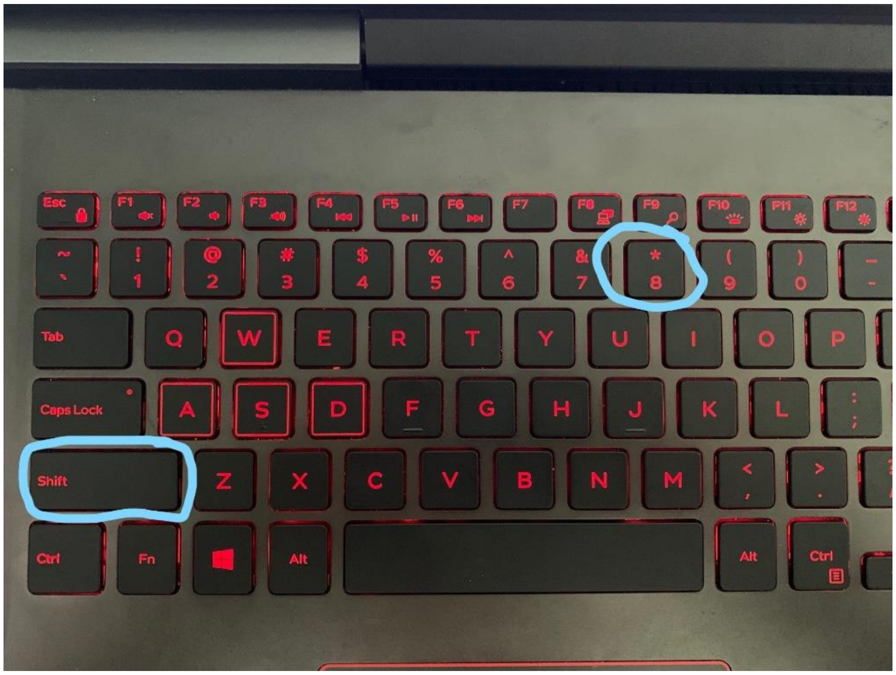
The output for this code should look something like this:



```
> clang-7 -pthread -lm -o main main.c
> ./main
Please enter an integer: 8
Please enter another integer: 5
8 - 5 = 3> 
```

Use the skills learned about addition and subtraction in programming to move on to multiplication and division.

In order to do multiplication in *C* we use the '*****' key which is done by holding down the shift key and hitting the 8 on the top of the keyboard while still holding the shift key, pictured here.



This will allow us to use multiplication which will look like this:

Sum = num1 * num2;

Now for division we will use '/' which is located next to the arrow keys on the keyboard and will look like this in code:

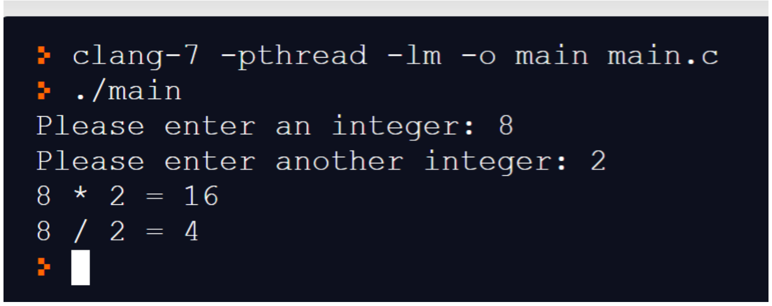
Sum = num1 / num2;

When doing division, we will want to make the variable that we wish to hold our answer in a Double type variable to hold decimals. This can be done by using the code shown in the data type table that was shown previously or by using this code:

double sum = 0;

Now try it out using what you learned before to multiply and divide 2 numbers to get the answer.

Your output should be similar to this:



```
> clang-7 -pthread -lm -o main main.c
> ./main
Please enter an integer: 8
Please enter another integer: 2
8 * 2 = 16
8 / 2 = 4
>
```

Part 3

You have now learned to do simple math in *C.*

There only one last thing you need to learn to complete this lesson and that is how to do slightly more advanced equations using parentheses. Do you remember that small equation from earlier? $x = \frac{(9*10)+2}{12-8}$

That’s the one, well this equation is too complicated as it is now for a computer to do so if we want to find the answer, we will need to split up parts of the equation into parts that a computer can solve. We do this using parenthesis as such:

$$x = (9 * 10) + \frac{2}{12} - 8$$

This is what the computer would see if we put the equation in as it is written so use parentheses to make this easier, first let’s do the top of the fraction:

$$x = \frac{((9 * 10) + 2)}{12} - 8$$

Now that you have closed the top in parentheses this is what the computer will see but we still want to subtract 8 from 12 on the bottom of the fraction so let's do the bottom now:

$$x = \frac{((9 * 10) + 2)}{(12 - 8)}$$

This is what the computer will see when all the parts of the equation are closed in parentheses now in code it will look like this:

```
X = ((9*10) +2) / (12-8);
```

Assessment

1. Apply the steps in this lesson to a simple math equation and create a small program to solve it.
2. Use the integer 7 so that the sum = 15
3. Use the integer 9 so that the sum = 3
4.

Additional Resources

- https://www.w3schools.com/
- https://www.geeksforgeeks.org/c-programming-language/
- https://replit.com/languages/c

Created By:
Ryan Floyd, Computer Science, '22

Thank you for downloading this lesson, please take a moment to complete our survey