# Introduction

Welcome to your favorite day of the week which is programming day🎉. This week, we shall work together to learn and implement new programming concepts.

### Skills to Learn:

* Distinguish between Local and Global Variables
* Move the Pacman using the Arrow Keys

### 

### Let's do some coding.

**Skill**: Distinguish between Local and Global Variables

## Introduction

So far, we have learned about various kinds of variables depending on datatypes such as int, float, char, etc. However, in a broader sense, the variables can be categorized into two major categories.

1. Local Variables
2. Global Variables

**Local Variables**

Local variables can be accessed inside the functions where they have been declared. These variables can only be accessed inside that function.

|  |  |
| --- | --- |
| Consider the following example: | |
|  |  |
| Here is the other scenario for you to consider: | |
|  |  |
| The above-mentioned examples define that the variables declared inside the body of a function cannot be accessed inside the body of other functions.  Such variables are called **Local Variables**. | |

**Global Variables**

Global variables can be accessed anywhere in the program and are not limited to a function only. For example:

|  |  |
| --- | --- |
| Consider the following example: | |
|  | Can you tell why there is no output? |
| Here is the other scenario for you to consider: | |
|  |  |
| The above-mentioned examples show that variables declared outside the bodies of functions can be accessed anywhere in the program and therefore are often referred to as **Global Variables**. | |

**Question!**

Can you guess the output of the below-mentioned code snippets?

**Task 01(CA):** Write the below-mentioned program to verify whether your prediction is true or false.

|  |  |
| --- | --- |
|  |  |
|  |
|  |

## Conclusion

|  |  |
| --- | --- |
| **Variables** | **Description** |
| Local Variables | This type of variable can only be **accessed only** inside the function(s) where they have been declared. |
| Global Variables | This type of variable can be **accessed anywhere** in the program. |
| **Congratulations !! You just have learned another skill.** | |

**Task 02(CP):**

Create a function that takes the length, width, height (in meters) and output unit in which you want to see the answer and returns the volume of a pyramid in the correct unit.

**Notes:**

* The units used are limited to: millimeters, centimeters, meters and kilometers.
* Ensure you return the answer and add the correct unit in the format cubic <unit>.

**Test Cases:**

* pyramidVolume(4, 6, 20, "centimeters") ➞ "160000000.000 cubic centimeters"
* pyramidVolume(1843, 1823, 923, "kilometer s") ➞ "1.034 cubic kilometers"
* pyramidVolume(18, 412, 93, "millimeters") ➞ "229896000000000.000 cubic millimeters"

**Task 03(CP):**

You’ve been hired by an Automobile company to write a program to help the tax collector calculate vehicle taxes. Vehicle taxes are based on two pieces of information; the price of the vehicle and the vehicle type code.



The formula of calculating the final price of an item is:

Final Price = Item Price + Tax Amount

Tax rates are in the table below.

|  |  |  |
| --- | --- | --- |
| **Vehicle Type** | **Vehicle Code** | **Tax Rate** |
| Motorcycle | M | 6% |
| Electric | E | 8% |
| Sedan | S | 10% |
| Van | V | 12% |
| Truck | T | 15% |

After the tax has been calculated, the program should display the following on the screen;

The final price on a vehicle of type xxx after adding the tax is $xxx.

with xxx replaced by the vehicle type and $xxx with the final price.

Your job is to write a function  
float taxCalculator(char type, float price);  
and then write the main function for taking the input from the user and then displaying the final output.

**Task 04(CP):**

A firm gets a request for creating a project for which a certain number of hours are needed. The firm has a certain number of days. During 10% of the days, the workers are being trained and cannot work on the project. A normal working day is 8 hours long. The project is important for the firm and every worker must work on it with overtime of 2 hours per day.

The hours must be rounded down to the nearest integer (for example, 6.98 hours are rounded to 6 hours).

Write a program that calculates whether the firm can finish the project on time and how many hours more are needed or left.

**Input Data**

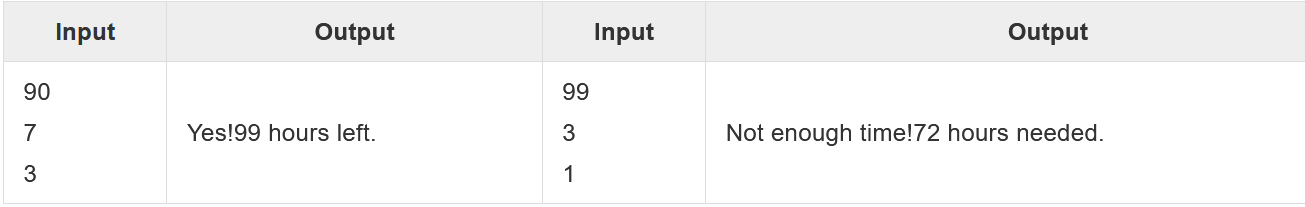
The input data is read from the console and contains exactly three lines:

* On the first line are the needed hours – an integer in the range of [0 … 200 000].
* On the second line are the days that the firm has – an integer in the range of [0 … 20 000].
* On the third line are the number of all workers – an integer in the range of [0 … 200].

**Output Data**

Print one line on the console:

* If the time is enough:
  + "Yes!{the hours left} hours left.".
* If the time is NOT enough:
  + "Not enough time!{additional hours} hours needed.".

****

**Skill:** Move the Pacman using the Arrow Keys

Last week, we made the Pacman patrol horizontally and vertically using the learned skills. Let’s now control that movement using the Arrow Keys of the Keyboard.

**Task 01(WP):** Control the movement of the Pacman using the Up, Down, Right, and Left arrow keys of the keyboard.

|  |
| --- |
|  |
| We can use the pre-defined function **GetAsyncKeyState()** by **Windows.h** library to capture the key pressed on the keyboard.  Following are the function calls that will be used in the program   |  |  | | --- | --- | | GetAsyncKeyState(VK\_UP) | Up Arrow Key | | GetAsyncKeyState(VK\_DOWN) | Down Arrow Key | | GetAsyncKeyState(VK\_LEFT) | Left Arrow Key | | GetAsyncKeyState(VK\_RIGHT) | Right Arrow Key | |
| Can you write the steps to solve this problem from here? |
| Following are the steps that need to be performed repeatedly.   * First, **spawn** the Pacman at the start of the screen using the gotoxy() function. * Detect which **direction key** is pressed from the keyboard. * Detect which **object** is present in that direction so   + if it is the **maze** that Pacman **should not move**   + if it is **not the maze** then **move** Pacman |
| Students recall how we move the Pacman in the previous lab |
| Each of the steps can be converted into a function.   * Clear Function to Print Blank space on the screen * Print function to print Pacman on the screen. |
| Let's sum it all up. **We need the following functions to perform respective tasks.**   1. getCharAtxy(x,y) 2. gotoxy(x,y) 3. erase(x,y) 4. printPacman() 5. printMaze() |

Following are the **function prototype** as derived from the above-mentioned discussion.

|  |
| --- |
|  |

Following are the **function definitions** of the above-mentioned functions.

|  |
| --- |
| We will use the same **gotoxy** function that we used in the last class. |
| We can the following function definition of **getCharAtxy** to check the character from the desired location on the console. |
| **erase** function will print a blank space on the screen. |
| **printPacman** will print the Pacman on the screen |
| **printMaze** will print the maze. |
| Now, let’s put all this logic into the **main()** function |
|  |
|  |
|  |
|  |
|  |
|  |
| Alright, now let’s compile and execute this program to check if we can control the Pacman inside the maze. |
| **Congratulations, You have learned how to control the Pacman inside the maze while tracking the score. That’s another skill added to your skillset !!** |

**Good Luck and Best Wishes !!**

**Happy Coding ahead :)**