Chapter 5: Conditionals

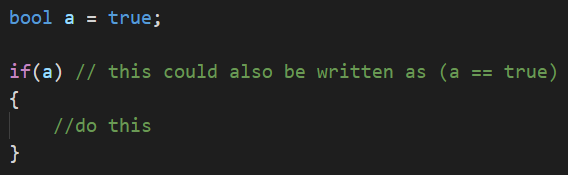
At some point in every program you will need to run different portions of code. For example, in a text adventure game the player will decide which direction to take at a fork in the road. We need some way of telling the program to execute a different portion of code depending on this choice. Unsurprisingly C++ has functionality built in for this type of decision-making.

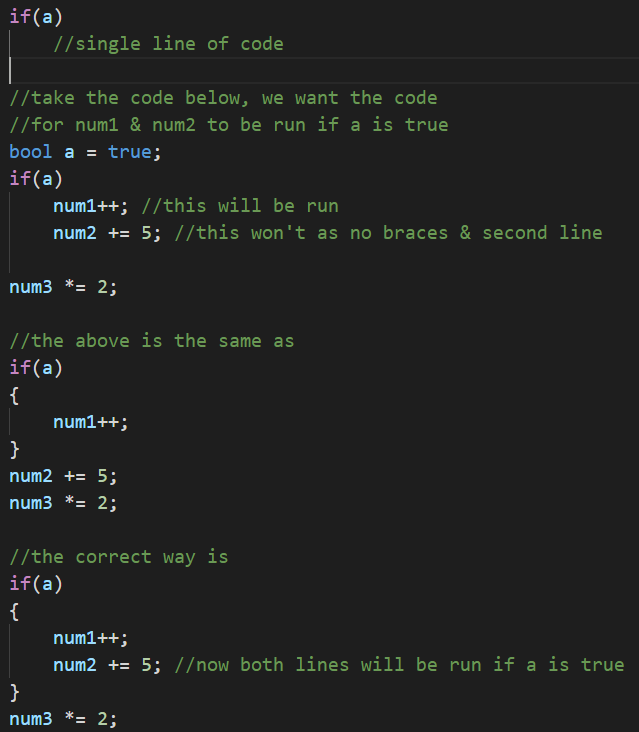
The following table shows the possible statements, which can be used. These will be used with the operators you looked at in the previous chapter. Following on from Table 5.1: Conditional Statements there will be an example depicting each approach in code.

|  |  |
| --- | --- |
| **Statement** | **Description** |
| if statement | An if statement consists of a Boolean expression followed by one or more statements. |
| if…else statement | An if statement can be followed by an optional else statement, which executes when the Boolean expression is false. |
| nested if statements | You can use one if or else statement inside another if or else statement. |
| switch statement | A switch statement allows a variable to be tested for equality against a list of values. |

Table 5.1: Conditional Statements

**Example of an if statement**

An if statement is the simplest form of condition check. It literally works, as you would expect – If the condition is true, do the following portion of code. If the condition equates to false, then the following portion of code is skipped. The portion of code to be executed is defined by the curly braces.

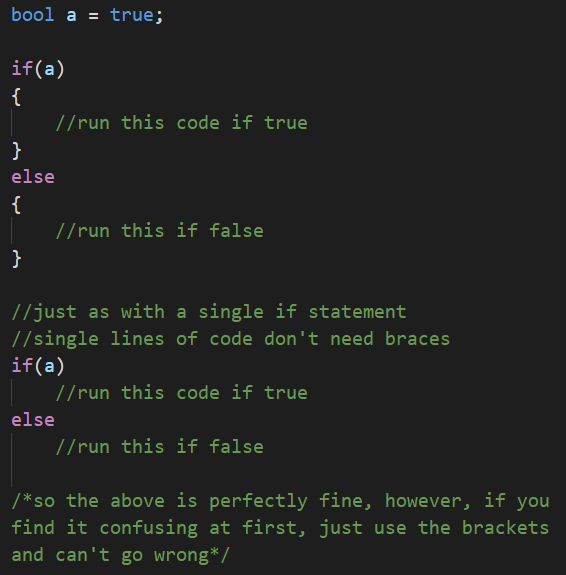
We can further reduce the code in the above example by removing the curly braces. This will only work if the code to be executed is a single line of code, this done in a similar manor to that of Python. If there is more than one line of code, only the first will be executed. The following lines will be executed after the if statement has exited. This may cause issues later as a syntax error will not be given. This is a logical error and the compiler cannot help you. 

**Task 1: Submit the answer as a screenshot in your class Notebook**

If num1 = 12; num2 = 5; and num3 = 7; what do they equal after the above sums are performed on them? **Num1 = num2 = num3=**

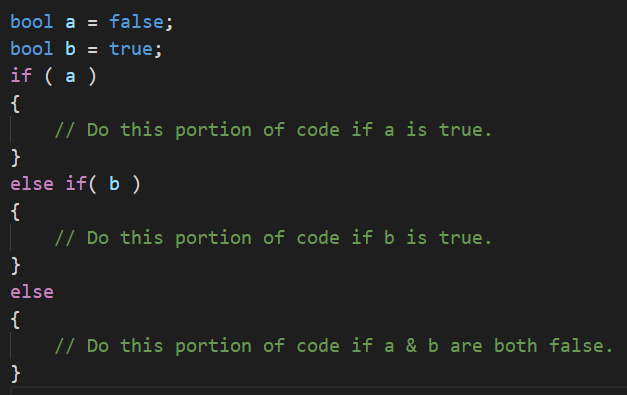
**Example of if – else statement**

We can extend an if statement to include a portion of code to be executed in the event of the condition being false. In English we are saying, if the condition is true do the first portion of code, otherwise do the second portion of code. This extra portion of code will only ever be executed if the condition is false.



**Example of if – else if – else**

Building on from the if…else statement, we can add another condition to the second portion of code. Here we are saying, if the first condition is true then do the first portion of code, otherwise check if the second condition is true, and if so do the second portion of code, and if neither are true, do the third portion of code.



Note: If a condition equates to true, the corresponding code portion will be executed. All others will be ignored.

**Task 2:**

Sticking with num1 = 12; num2 = 5; and num3 = 7; Set a to equal false, b to equal true and add the following scenarios:

If a true – num1 += 10; num2 += 15;

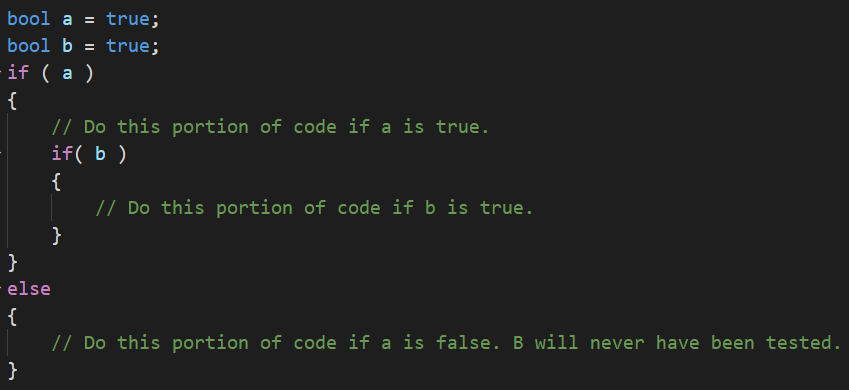
If b true – num1++; num3 += 20;

If both false – num1--; num2--; num3--;

Screenshot the output to class Notebook.

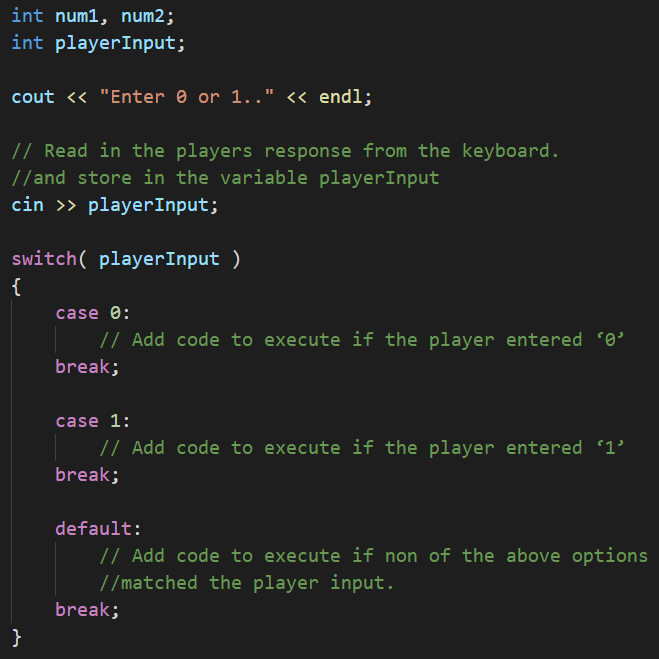
**Nested if statements**

Any legal code can go inside the if statement portion of code to be executed; therefore, it is possible to have any of the above variants of an if statement inside an if statement code block.



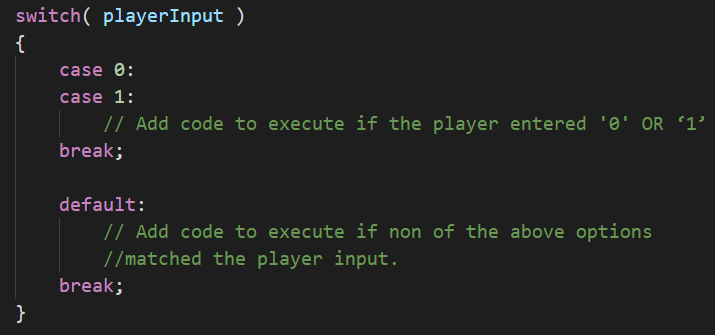
**Switch statements**

When you have a lot of options multiple if statements can become messy. Take the text-based adventure example again. The player could enter numerous responses to the on-screen prompt and you as programmer would need to code an if statement for each one. Alternatively, you can use a switch statement. A switch statement allows a variable to be tested against a list of constant values.



As you can see from the above code snippet, the variable to be tested against goes into the brackets beside the switch keyword, then within the curly braces are a list of values to compare against. We are saying, in the case of playerInput matching with ‘0’ do this portion of code. It is crucial to include the break keyword, otherwise code execution will automatically drop through to the next section without checking the condition.

The last part of a switch statement to note is the **default** case. This states that if none of the values match then do this portion of code. It acts in the same way as an else portion of an if…else statement.

There may be occasions where you wish a portion of code to execute if several conditions are met, in the same manner as an if statement which uses the Logical OR operator. In this case follow the below format.

It is also possible to nest switch statements and is a common occurrence in programming. This is done the same way as seen above with the nested if statements.

**Task 3:**

Building on what was covered in the slides and here, create a switch statement of chars.

Create a char variable called play\_again

Ask user if they want to play again, y or n?

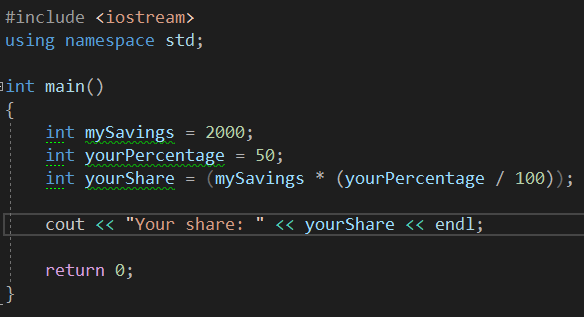
Store the char in play\_again and enter the switch statement. If case ‘y’ cout “you chose to play again”. If ‘n’, cout “goodbye” and if neither cout “that’s not y or n”

Test all conditions and paste outputs in class Notebook.

**Program 9: Share of Savings**

1. If you haven’t already open VS and start a new project called Project9\_ShareOfSavings

2. Replicate the following code:



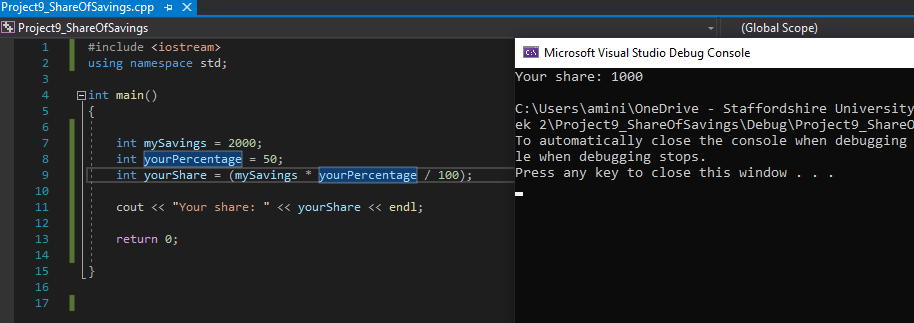
Run this program in Visual studio and see if you can understand why we get a wrong answer printed out instead of the correct answer of 1000. Then try and fix the error! Note there are multiple ways of fixing this and the quickest solution requires only 2 keystrokes!

Add the fixed code below along with a screenshot of the output.

**Program 9 Source code:**



**Program 9 Screenshot:**

****

**Program 10: Share of Haribo**

Pretend that, for some reason, you are now entitled to my stash of Haribo. There are 40 packets available and to make things fair, they are to be shared out equally, as in you all must have the same number of packets.

On the day when I decide to hand out the Haribo, only 14 students turned up.

**Write a program that**

1. Calculates how many of the 40 packets of Haribo do each of the students get?

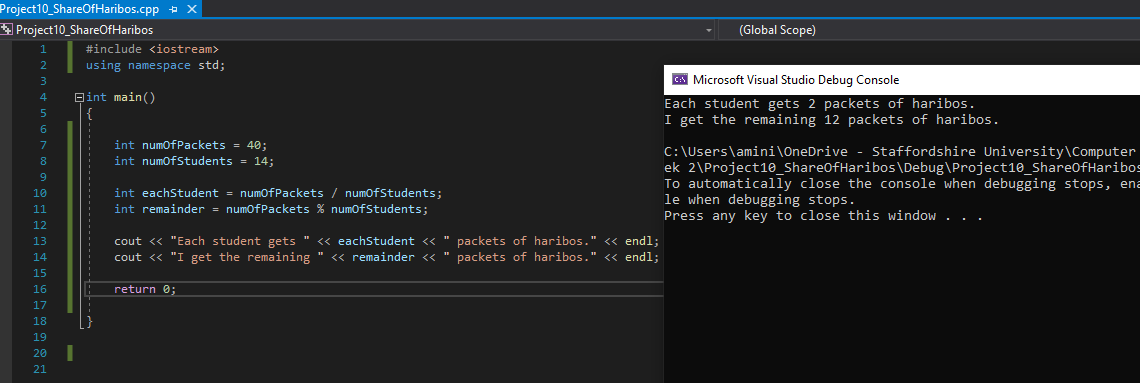
2. Calculates how many I get, which will be the remainder left after sharing the 40 packets as equally as possible

Your final answers should be **2** packets for each student, leaving **12** packets left for me. What we want to see is you coding the solution and getting the computer to do the right calculations that come up with these 2 answers. We are not bothered about the answers themselves, like how many mathematics questions are more concerned with “method marks” than the final answer.

**Program 10 Source code:**



**Program 10 Screenshot:**



**Program 11: Weather Menu**

Write a program that displays a short menu such as:

o Please choose an option: 1. Sunny 2. Cloudy 3. Raining 4. Exit

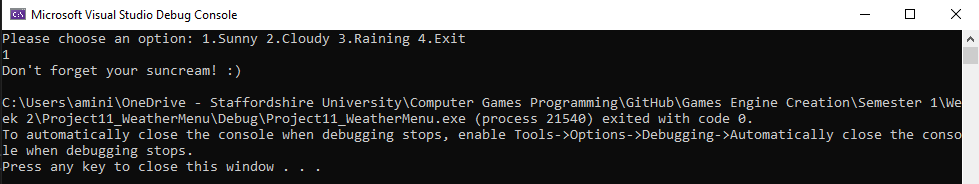
The program should use a **switch statement** to display a short message appropriate to the option chosen (such as "Don't forget your sunscreen").

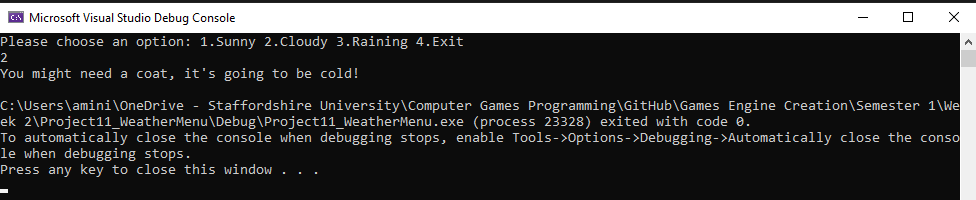
For an example of reading input from the player, see the switch statements example above and the use of **cin**.

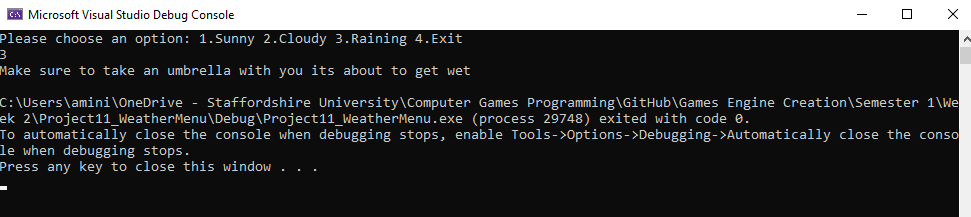
**Program 11 Source code:**

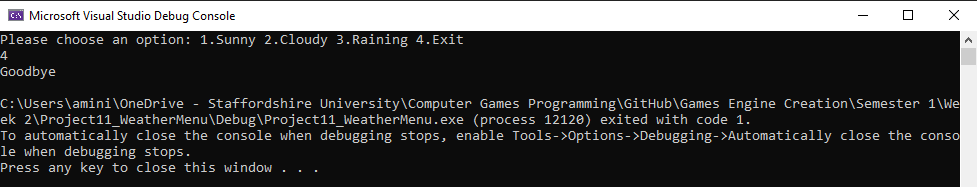
****

**Program 11 Screenshot:**

****

****

****

****

**Program 12: If This Then That**

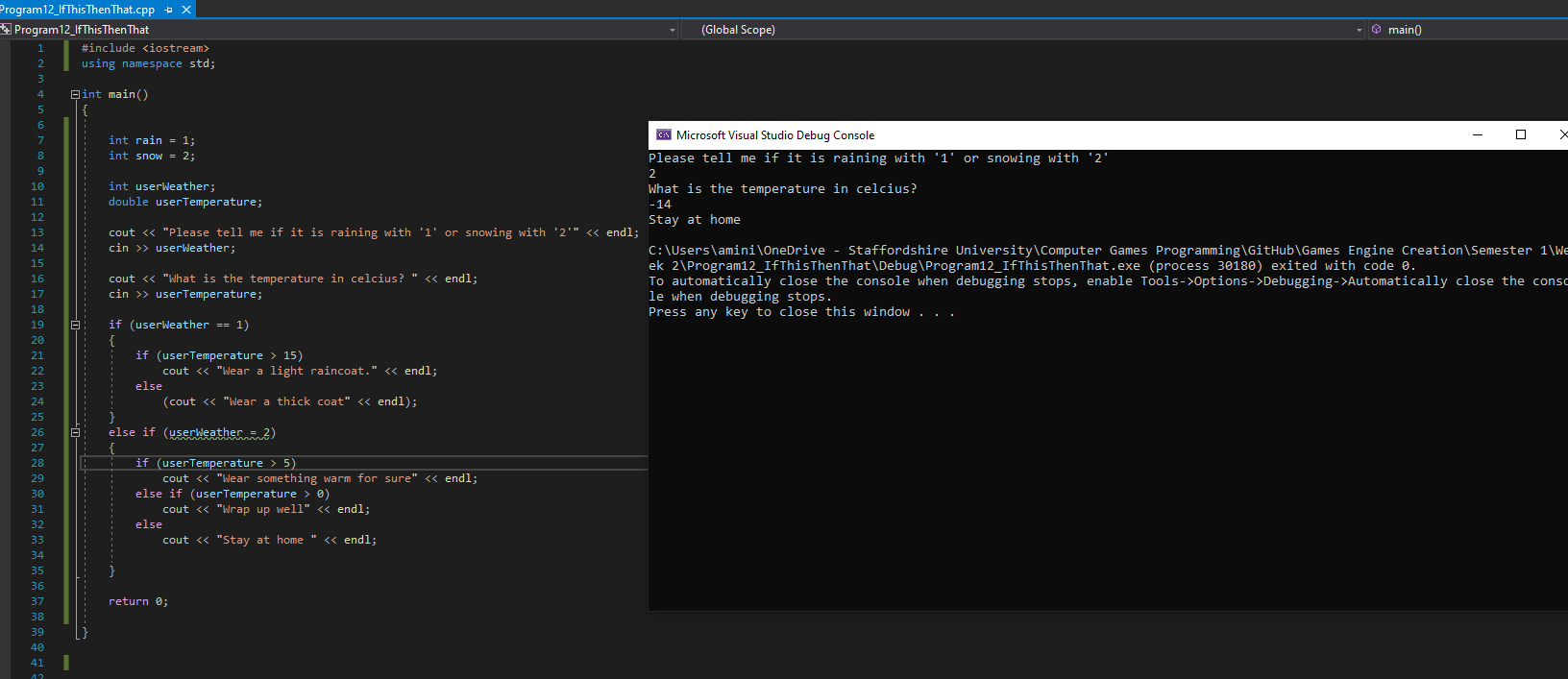
This program is going to be a little more complex than the previous programs. For this task you will need to create a program that does the following:

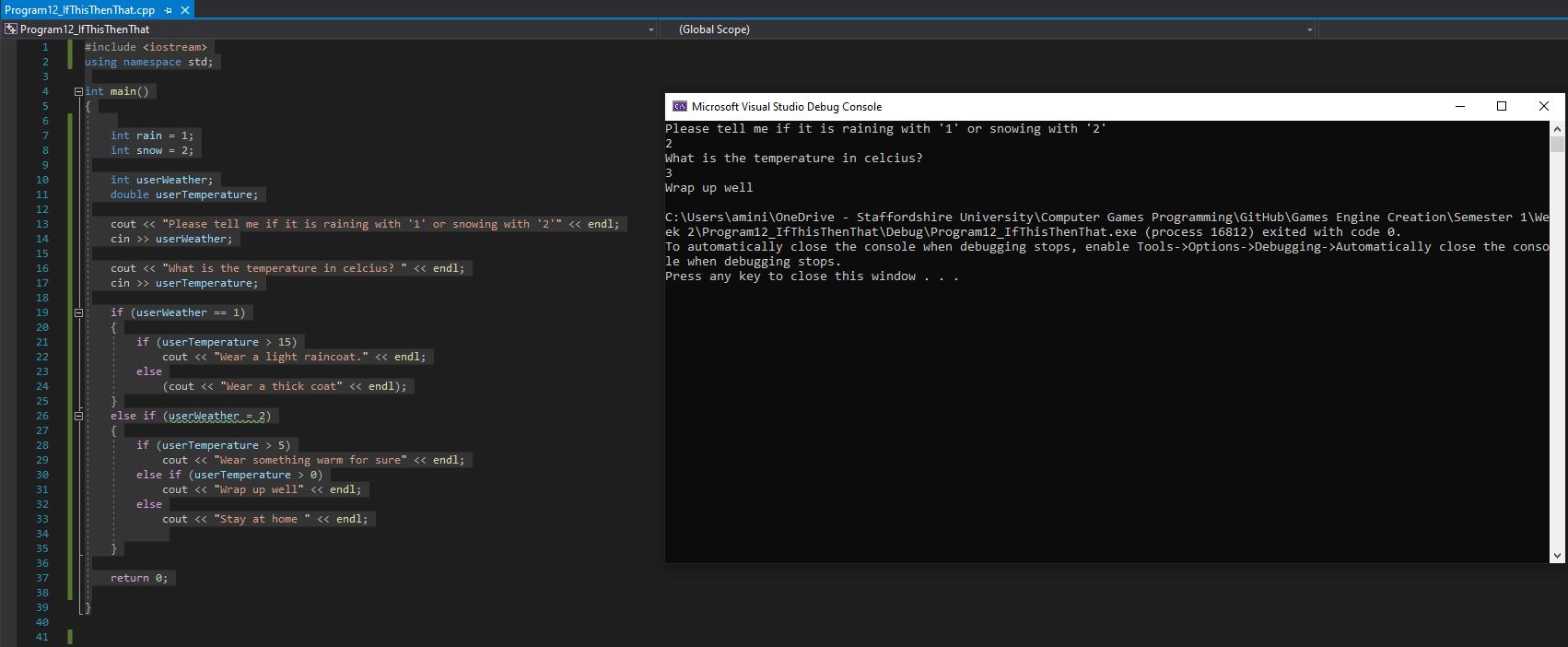
* Stores two ints, rain = 1; and snow = 2.
* You will need to create two more variable to hold a user’s answers and then ask the following:
  + Please tell me if it is raining with ‘1’ or snowing with ‘2’. And store the answer.
  + Then ask what the temperature is in Celsius.
* Next create an if – else if – else statement
  + If the users answer is equal to 1, jump into a nested **if else** statement with the condition of if temp > 15, wear a light raincoat, otherwise wear a thick coat.
  + If the user selected 2, jump into a **nested** if – else if – else statement with the conditions of, if temp > 5 tell them to wear something warm. Or if temp > 0 tell them to wrap up well, otherwise tell them to stay home.
  + The final outer else should just inform them to have a nice day as it is neither raining nor snowing.
* Test all your conditions to make sure your program works correctly.

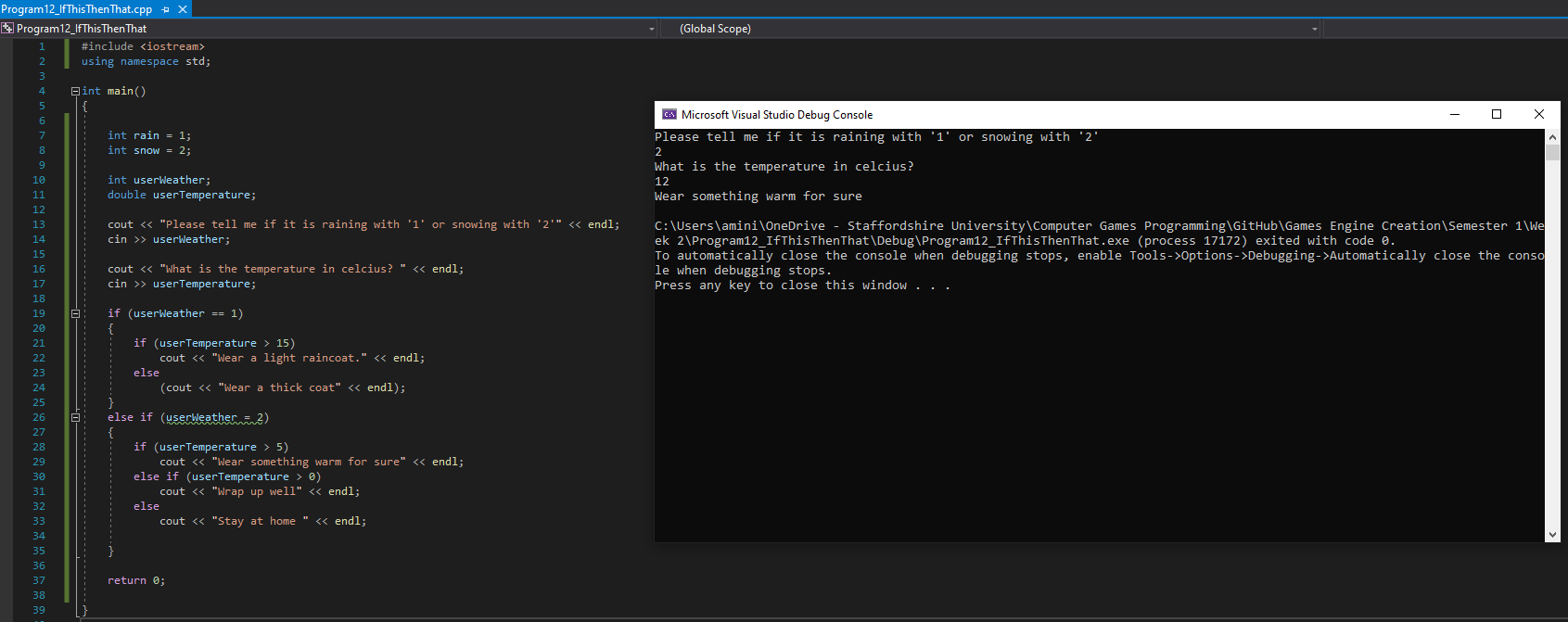
**Program 12 Source code:**

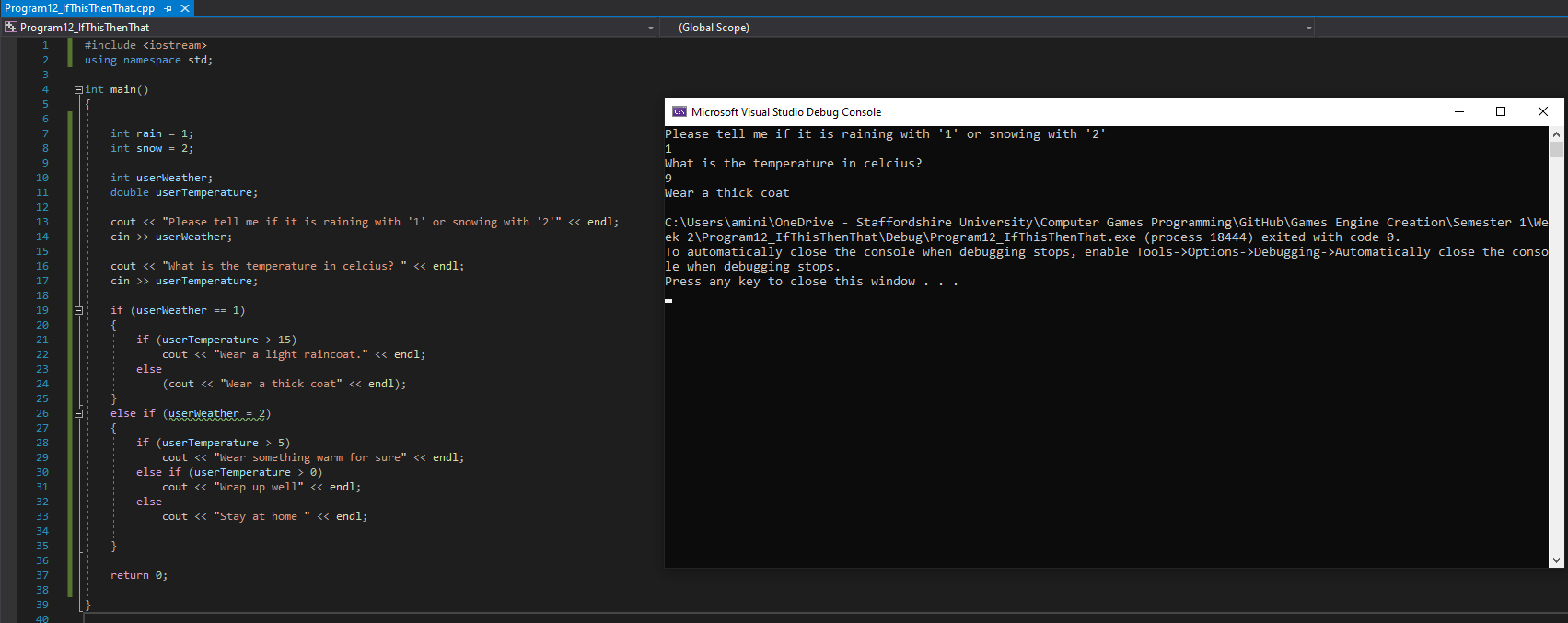
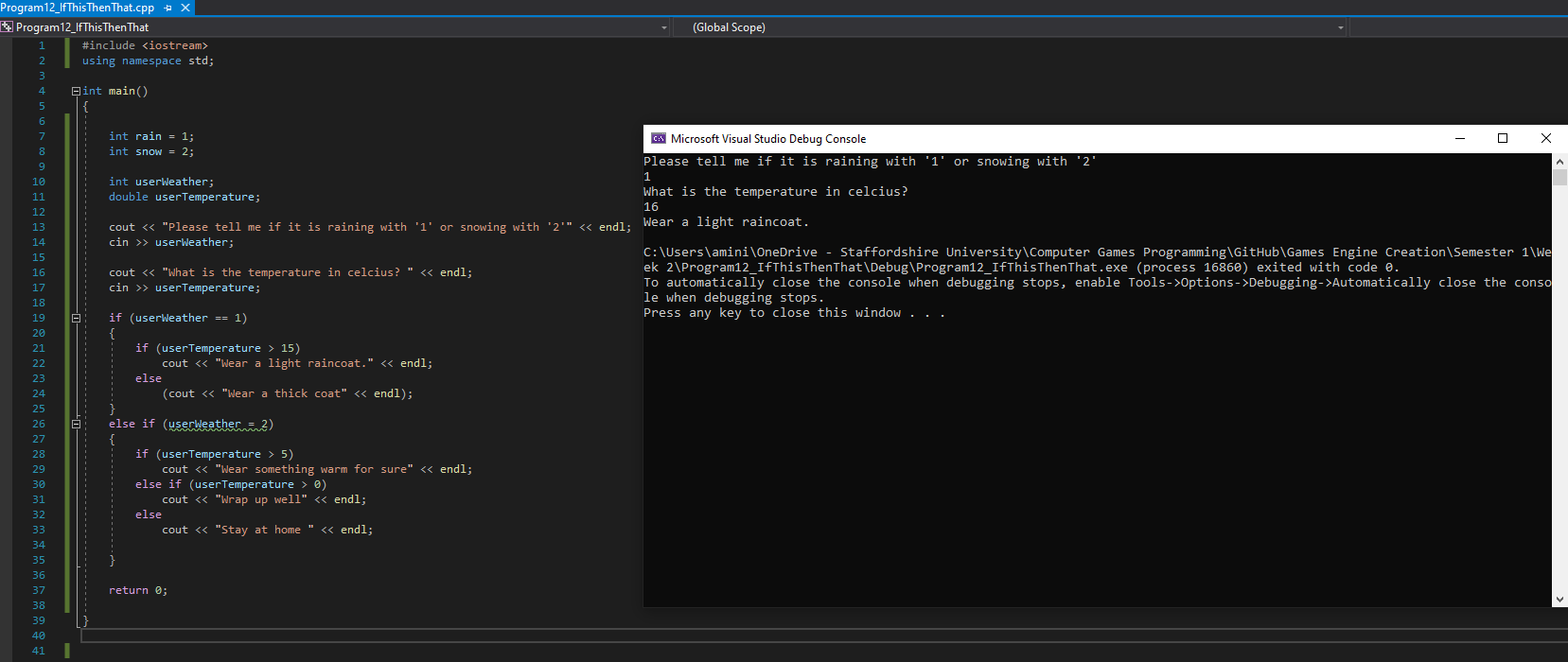
****

**Program 12 Screenshot:**

****

****

****



**Program 13: Discount**

For this task you will be creating a program with an if – else if – else statement that has multiple conditions. It must have the following:

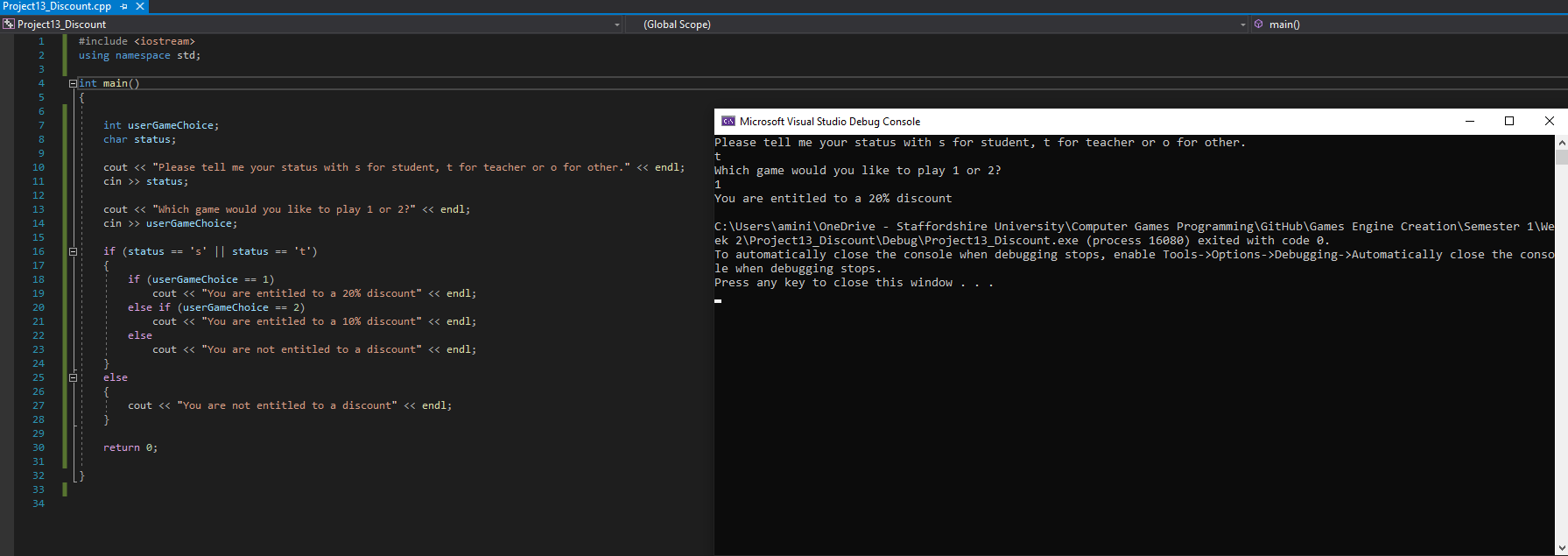
* A char to hold/store the user’s status and an int for the user’s game choice.
* You will then need to ask:
  + Please tell me your status with ‘s’ for student, ‘t’ for teacher or ‘o’ for other.
  + Then ask which game they would like, 1 or 2?
* Next create an if – else if – else statement
  + If the user’s status is equal to **s** or **t** and the game value is equal to **1**. Output a message that they can get 20% discount.
  + If the user’s status is **s** or **t** and the game value is equal to **2**. Output a message saying they can get a 10% discount.
  + Otherwise inform them they are not entitled to a discount.
* Test all your conditions to make sure your program works correctly.

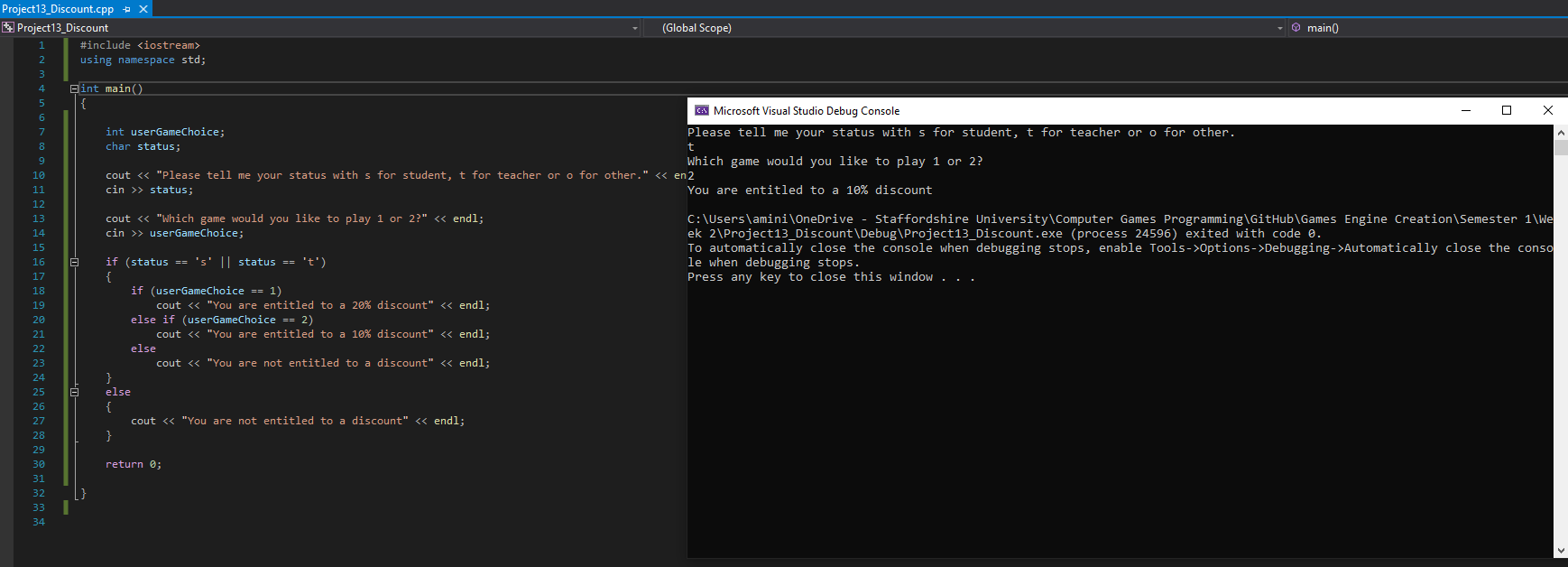
Hint: If you are struggling refer to the slides in the PowerPoint for help. Read the instructions carefully, writing pseudocode may help.

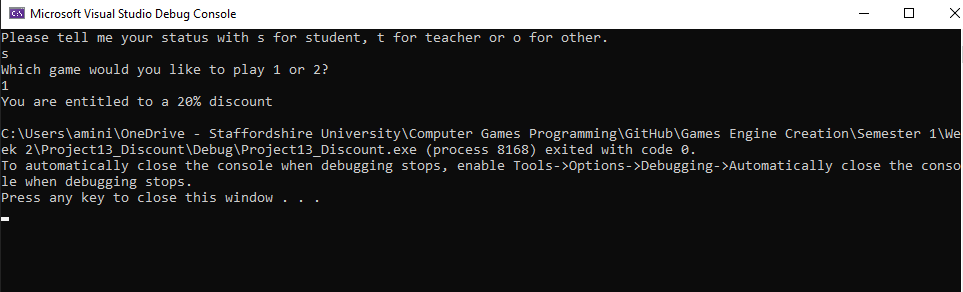
**Program 13 Source code:**

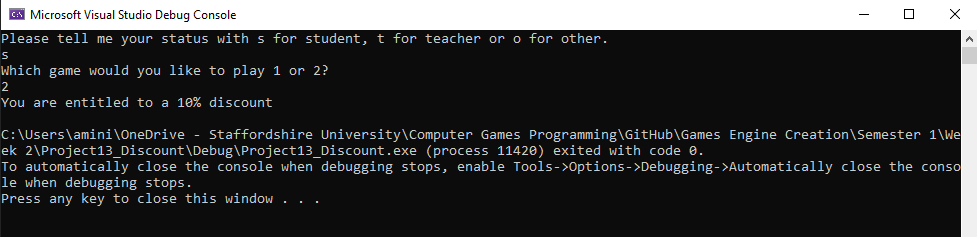
****

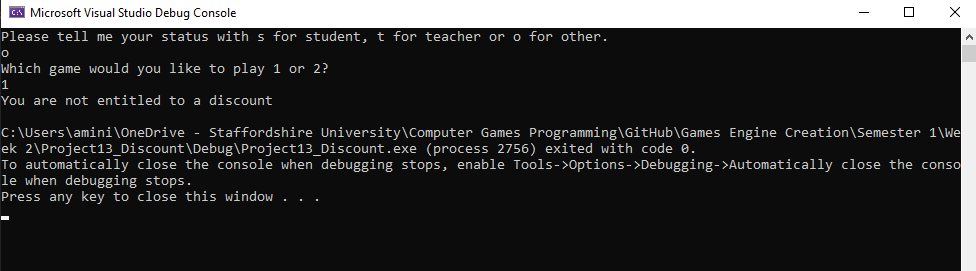
**Program 13 Screenshot:**

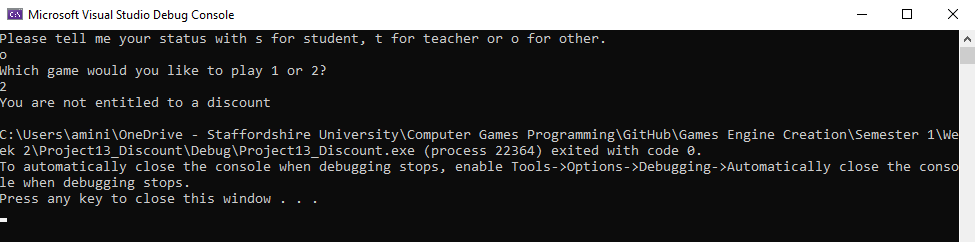
****

****

****

****

****

****