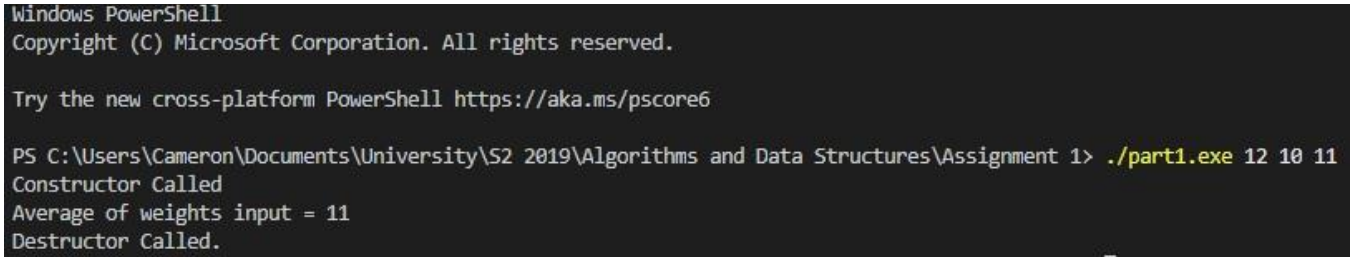
CSC2401
Algorithms and Data Structures
Assignment 1

Author: Cameron Duncan-Kemp
Student ID: u1032841

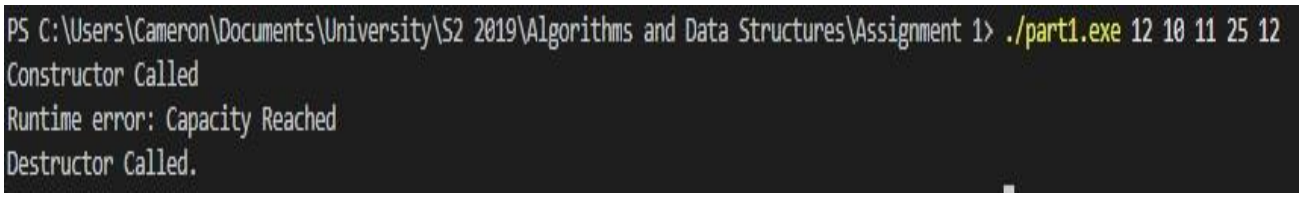
Part 1 Testing:

Normal Operation:



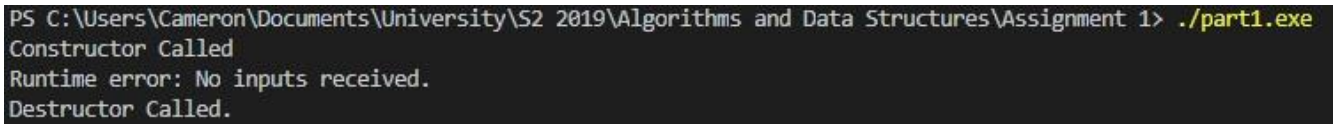
The constructor was called and announced, the correct average weight has been calculated and displayed. After this has been completed, the destructor is announced and called, which then proceeds to delete the object before terminating the program.

Too Many Samples:



With this execution of the program, whilst both the constructor and destructor are called and announced properly, the average is not displayed. In place of this, a runtime error has been caught and displayed showing that the number of arguments received has exceeded the capacity of the program's array.

No Inputs:



With this execution of the program, whilst both the constructor and destructor are called and announced properly, the average is not displayed. In place of this, a runtime error has been caught and displayed showing that no input arguments have been received. As it is impossible to divide by zero, it throws the runtime error instead.

Part 2 Testing:

Normal Operation:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part2.exe
Constructor Called
Please input a weight, input 0 to finish: 12
Please input a weight, input 0 to finish: 18
Please input a weight, input 0 to finish: 45
Please input a weight, input 0 to finish: 0
Average of weights input: 25
Destructor Called.
```

From normal operation testing, the program has displayed that it has called both the constructor and destructor for the sampleWeight object. After receiving three valid inputs; 12, 18 and 45, the terminating input of 0 is given and the program gives the correct average of the three valid inputs which is 25.

No Inputs:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part2.exe
Constructor Called
Please input a weight, input 0 to finish: 0
Destructor Called.
Domain error occured: No inputs received.
```

When the program is given only the terminating input of 0, it still calls both the constructor and destructor for the sampleWeight object as the object is still created before it is ready to receive the user input, and as such still needs to be destroyed as well. As no valid inputs were received, the domain error was thrown and caught informing the user that no inputs were received for the execution of the program.

Too Many Samples:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part2.exe
Constructor Called
Please input a weight, input 0 to finish: 10
Please input a weight, input 0 to finish: 156
Please input a weight, input 0 to finish: 14
Please input a weight, input 0 to finish: 15
Please input a weight, input 0 to finish: 12
Destructor Called.
Domain error occured: Capacity Reached
```

During this execution of the program, after the initial object constructor has been called, 5 user inputs were received, as the maximum allowed by the program is 4 inputs, the program terminates without needing the terminating input of 0 to be entered by the user but instead does it after receiving the 5^{th} input. After this offending input is entered, the destructor gets called to delete the object itself and a domain error is thrown and caught displaying to the user that the capacity for the program has been reached.

Part 3 Testing:

Normal Operation:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part3.exe
Constructor Called
Please input a weight, input 0 to finish: 10
Please input a weight, input 0 to finish: 25
Please input a weight, input 0 to finish: 13
Please input a weight, input 0 to finish: 0
Average of weights input: 16
Destructor Called.
```

During normal operation, when the program received three valid inputs; 10, 25 and 13, it called the constructor correctly and proceeded to output the correct average weight of 16 before calling the destructor to free the resources held by the variables within the program. No memory leak has occurred.

No Inputs:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part3.exe
Constructor Called
Please input a weight, input 0 to finish: 0
Domain error occured: No inputs received.
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1>
```

When the program received no valid inputs, only the terminating input of 0, the constructor was called as it is set to do so before receiving inputs from the user, it then proceeds to throw the domain error detailing that it has not received any valid inputs. The program does not execute the destructor as it is currently using a pointer to allocate the memory for the object dynamically, which means that it would not be triggered to automatically call the destructor when the object would normally go out of scope when the exception error terminates the program. Since the exception error terminates the program early, it also does not proceed to the delete command within the program to initiate the destructor for the object. In this example, a memory leak has occurred as the resources taken by the object pointer were not deallocated by the destructor.

Too Many Samples:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part3.exe
Constructor Called
Please input a weight, input 0 to finish: 123
Please input a weight, input 0 to finish: 15
Please input a weight, input 0 to finish: 12
Please input a weight, input 0 to finish: 16
Please input a weight, input 0 to finish: 12
Domain error occured: Capacity Reached
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1>
```

With this execution of the program, it does initially call the constructor before prompting the user for input, when it received the 5th input, it threw the domain error noting that it had reached full capacity. Much like the previous test, the destructor is not called, and the exception error terminates the program early, it also does not proceed to the delete command within the program to initiate the destructor for the object. In this example, a memory leak has occurred as the resources taken by the object pointer were not deallocated by the destructor.

Part 4 Testing:

Normal Operation:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part4.exe
Constructor Called
Please input a weight, input 0 to finish: 10
Please input a weight, input 0 to finish: 15
Please input a weight, input 0 to finish: 13
Please input a weight, input 0 to finish: 0
Average of weights input: 12.6667
Destructor Called.
```

During normal operation, when the program received three valid inputs; 10, 15 and 13, it called the constructor correctly and proceeded to output the correct average weight of 12.6667 before calling the destructor to free the resources held by the variables within the program. No memory leak has occurred.

No Inputs:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part4.exe
Constructor Called
Please input a weight, input 0 to finish: 0
Destructor Called.
Domain error occured: No inputs received.
```

Unlike the previous no-input test, this time the program successfully called both the constructor and destructor despite throwing the domain error indicating that it has received no inputs. Due to the program using the unique_ptr for the object, it is automatically deleted after it goes out of scope. No memory leak has occurred.

Too Many Samples:

```
PS C:\Users\Cameron\Documents\University\S2 2019\Algorithms and Data Structures\Assignment 1> ./part4.exe
Constructor Called
Please input a weight, input 0 to finish: 5
Please input a weight, input 0 to finish: 13
Please input a weight, input 0 to finish: 17
Please input a weight, input 0 to finish: 13
Please input a weight, input 0 to finish: 12
Destructor Called.
Domain error occured: Capacity Reached
```

Unlike the previous too many samples test, this time the program successfully called both the constructor and destructor despite throwing the domain error indicating that it had reached the capacity of inputs. Due to the program using the unique_ptr for the object, it is automatically deleted after it goes out of scope. No memory leak has occurred.