Lab 07 - Hash Functions

Instructions:

- **The lab requires completing a few tasks.**
- **Your submissions must be submitted to the Lab07 directory of your GitHub repository or uploaded to the Lab07 assignment on Google classroom.**
- **Accompanying these instructions are a few header files that must be included in the appropriate programs you have to write.**
- **Besides the header files provided, your programs can only include the libraries *iostream*, *string*, *fstream*, *cstdlib*, *ctime*, *sstream*, and *cctype*.**
- **Cheating of any kind is prohibited and will not be tolerated.**
- **Violating and/or failing to follow any of the rules will result in an automatic zero (0) for the lab.**

TO ACKNOWLEDGE THAT YOU HAVE READ AND UNDERSTOOD THE INSTRUCTIONS ABOVE, AT THE BEGINNING OF YOUR SUBMISSION(S), ADD A COMMENT THAT CONSISTS OF YOUR NAME AND THE DATE

Grading:

Task	Maximum Points	Points Earned
1	2	
2	4	
3	4	
Total	10	

Note: solutions will be provided for tasks colored blue only.

Task 1

Create a file named Extra.h that defines a hash function named `Hash()` that will help with Task 2. The function must have a constant big-O runtime.

Task 2

Create a file named main.cpp that includes Extra.h and defines an int function named `MinMostRepeated()` whose header is

```
int MinMostRepeated(Array<int>& data)
```

Given that *data* is an array of three-digit integers, the function returns the value in *data* that is the minimum value of the set of numbers from *data* whose tens digit appears the most. If multiple sets from *data* contain the same maximum count, the minimum value of the maximum tens digit will be returned. For instance, the caller `MinMostRepeated([-123,864,281,133,-977,310,458,928,184])` will return 184 because 8 is the greater tens digit repeated the most and 184 is the minimum value in the set ($\{281, 184\}$). Furthermore, the function must have a linear big-O runtime and call `Hash()`.

Task 3

Copy the Project.h file from Lab 06 to Lab 07, and modify the file as follows

- **move `Preload()` into the class *Game* as a private method.**
- **modify the default constructor to create an string *Array* object, call `Preload()` with the string `"default.txt"` and the *Array* object as its arguments respectively. And then, only if `Preload()` returns a value other than 0, randomly assign one of the elements of the *Array* object to *board*. Otherwise, do not change the value of *board*.**
- **a public void method named `GetInputs()` that takes no parameters. It continuously prompts the user to enter two rows and two columns separately and stores the inputs until the input coordinates are valid.**