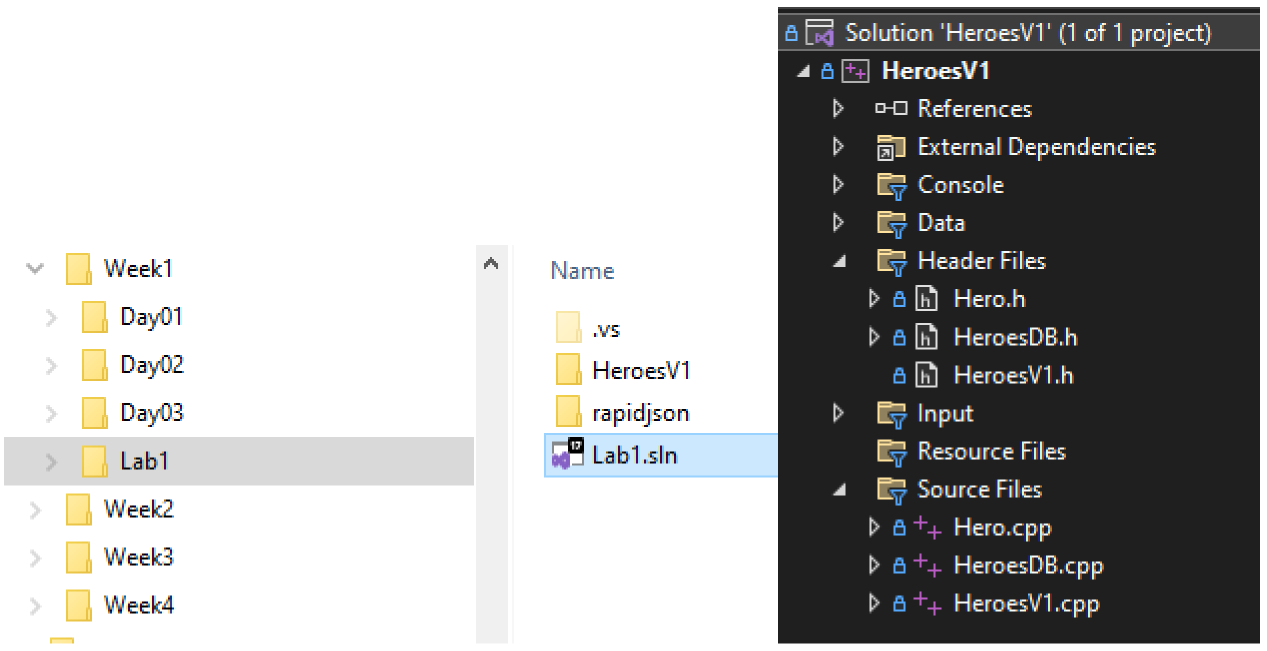
COP2334

PG2 – LAB: HEROES V1

CONTENTS

SETUP

A **C++ console application** has been provided for you in your **GitHub repo**. **Use the provided solution.**



Lab Video

Here's a video showing what the lab could look like when completed:

https://fullsailedu-my.sharepoint.com/:v:/g/personal/ggirod_fullsail_com/EWssuy9b6A9FkaK8aGZ541ABGXeezQFpu8pDFbOuQZg9AA?e=3tothg

How should you proceed?

The typical process for approaching the labs is...

Work on each part in order.

1. Watch the lecture video for Part A.
    a. The lectures have ***optional*** coding challenges. If you have time, then attempt them because they will help explain what you need to do for the lab.
2. Start working on Part A for the lab (Parts A-1 through A-3).
3. Once done with Part A, move on to Part B and repeat the steps above.

What to do if you need help?

Don't struggle for too long (more than 1 hour). Reach out for help to get your questions answered.

- Take advantage of the **open lab sessions** (they are in the evening so check the announcement channel in Discord for the zoom link). Here is the schedule: https://fullsailedu.sharepoint.com/:x:/s/emergingtechstu/EbBgblouNORLsms4V4hpAYgB0Mejv59w8IVGADkVgfJW8Q?e=0l2svR
- Ask your questions in the **PG2 Discord channels**. There will be instructors and students online who can help.
- Check the availability of **Full Sail tutors**. (https://discord.com/invite/8nV8PBqq7z)

Committing your code

You MUST commit and push your code to GitHub after completing each section. For instance, part A of this lab has 3 sections. You must commit and push after completing each section which would mean you end up with 3 separate commits for part A.

If you do not commit and push for every section, you could be deducted up to 15 points for the lab.

PART A

Lecture Videos for Part A

Day 01 Methods and Vectors Lecture:

https://fullsailedu-my.sharepoint.com/:v:/g/personal/ggirod_fullsail_com/EVIOtN0d2eBOq8nqN3OANdEBqs4zeY3I0WiRCP5DSsDL0g?e=cfMS56

Tips:

ACCESSING THE PARTS OF THE HERO CLASS:

If you have a Hero object, then to access any of the elements inside the object, you use the **dot operator**.

Example to show the Speed of the hero…

Hero myHero = _heroes[0]; //get the first hero in the vector

int speed = myHero.**Powerstats().Speed**; //use the dot operator to access the Speed value of the Powerstats() getter of the myHero variable.

METHODS

Creating and callings methods: C++ Class Methods (w3schools.com)

Parameters and arguments: C++ Function Parameters (w3schools.com)

Using returned values: C++ Functions - Return (w3schools.com)

VECTOR<T>

Working with the vector class: Vector in C++ STL - GeeksforGeeks

Helper Classes

There are 2 helper classes that are provided for the labs: **Input** and **Console**. You'll find the files under the **Misc/Input** and **Misc/Console** folders in the Solution Explorer.

THE INPUT CLASS

The Input class is provided in the labs to make it easier to get input from the user. You'll find the class in the labs in the **Misc/Input** folders in Solution Explorer. Investigate the **Input.h** file to discover the methods available and how to call them.

There are 4 methods: **GetString**, **GetInteger**, **GetMenuSelection**, and **PressEnter**. These methods are static therefore to call them you use the Input class name with the :: scope resolution operator.

Examples:

```
std::string myName = Input::GetString("What is your name?");

int age = Input::GetInteger("What is your age?", 0, 120);
```

Notes:

- the first parameter to these methods is the message to show to the user.
- The 2nd and 3rd parameters to the GetInteger method is the min and max range for the integer.

THE CONSOLE CLASS

The Console class is provided in the labs to make it easier to print things to the Console window. You'll find the class in the labs in the **Misc/Console** folders in Solution Explorer. Investigate the **Console.h** file to discover the methods available and how to call them.

Examples:

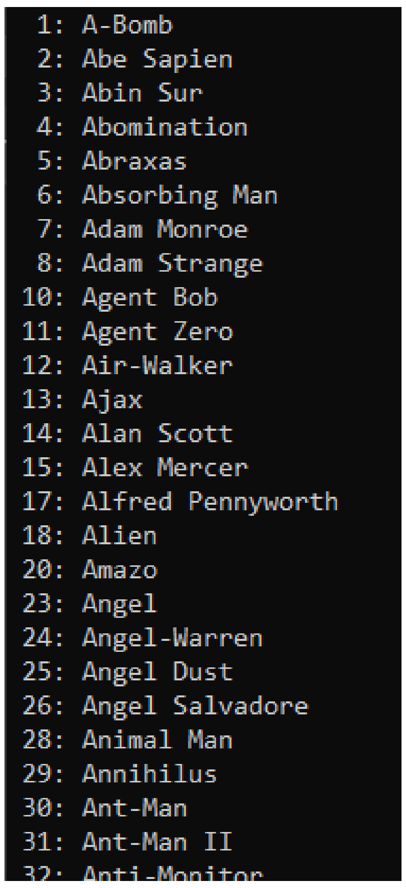
```
Console::Write("Hello Gotham.");
Console::Write("Hello Gotham.", ConsoleColor::Yellow, ConsoleColor::Cyan);
Console::WriteLine("Hello Gotham.");
Console::WriteLine("Hello Gotham.", ConsoleColor::Yellow, ConsoleColor::Cyan);
Console::SetCursorLeft(15);
Console::SetCursorPosition(5, 10);
```

Part A-1: ShowHeroes

Add a method called **ShowHeroes** to the **HeroesDB** class. **Declare your method in the HeroesDB.h file and define the method in the HeroesDB.cpp file (**C++ Class Methods (w3schools.com)**).** The method should loop over the _heroes vector and print the hero ID and the hero Name.

NAME	RETURNS	PARAMETERS	COMMENTS
ShowHeroes	nothing	nothing	Show the ID and name for each hero.

In **main** (which is in **HeroesV1.cpp**), add code to case 1 of the switch to call the method.



```
 1: A-Bomb
 2: Abe Sapien
 3: Abin Sur
 4: Abomination
 5: Abraxas
 6: Absorbing Man
 7: Adam Monroe
 8: Adam Strange
10: Agent Bob
11: Agent Zero
12: Air-Walker
13: Ajax
14: Alan Scott
15: Alex Mercer
17: Alfred Pennyworth
18: Alien
20: Amazo
23: Angel
24: Angel-Warren
25: Angel Dust
26: Angel Salvadore
28: Animal Man
29: Annihilus
30: Ant-Man
31: Ant-Man II
32: Anti-Monitor
```

Make sure to commit + push after completing the section.

Part A-2: RemoveHero

Add a method called **RemoveHero** to the **HeroesDB** class. **Declare your method in the HeroesDB.h file and define the method in the HeroesDB.cpp file (**C++ Class Methods (w3schools.com)**).** The method should have a string parameter for the name of the hero to remove. The method should loop over the heroes vector. If the hero is found, remove the hero from the heroes vector. Return true if the hero was found and removed. Return false if the hero was not found.

When checking for a match, **make sure to ignore the case** of the name and the parameter. Use the `_stricmp (MSDN Link)` method to compare strings. Note: you'll need to call c_str() on the std::string when calling the method.

An example of calling _stricmp:

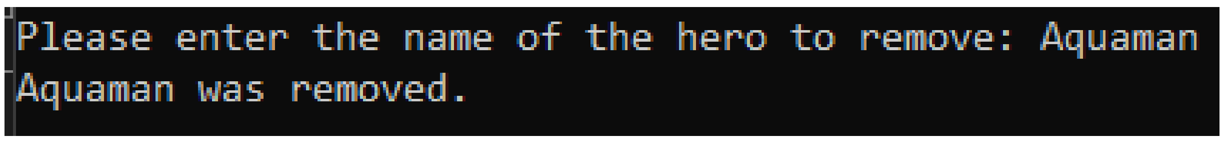
```
std::string s1 = "Batman", s2 = "Aquaman";
int compResult = _stricmp(s1.c_str(), s2.c_str());
```

_stricmp returns an int:

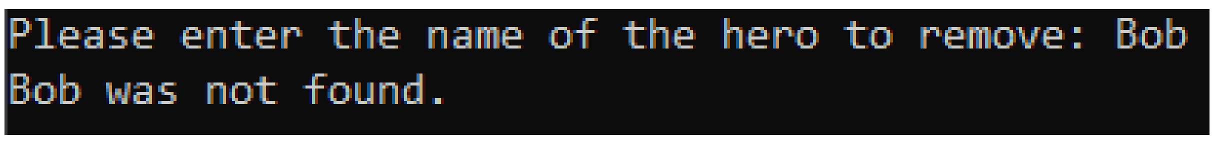
- < 0 if s1 is LESS THAN s2
- 0 if s1 is EQUAL TO s2
- > 0 if s1 is GREATER THAN s2

NAME	RETURNS	PARAMETERS	COMMENTS
RemoveHero	bool	string	Removes the hero from the heroes vector if it is found and returns true. If not found, returns false.

In main (which is in **HeroesV1.cpp**), add code to case 2 of the switch. Using **Input::GetString**, ask the user to enter the name of the hero to remove. Call **RemoveHero** passing the string that the user enters. **In main**, if the returned value is true, print that the hero was removed else print that the hero was not found.



```
Please enter the name of the hero to remove: Aquaman
Aquaman was removed.
```



```
Please enter the name of the hero to remove: Bob
Bob was not found.
```

Make sure to commit + push after completing the section.

Part A-3: StartsWith

Add a method called **StartsWith** to the **HeroesDB** class. **Declare your method in the HeroesDB.h file and define the method in the HeroesDB.cpp file (**C++ Class Methods (w3schools.com)**).** The method should have a string parameter for the name of the hero to match. Loop over the heroes vector and add every hero whose name **starts with** the string parameter to a vector. Return the vector.

When checking for a match, **make sure to ignore the case** of the name and the parameter.

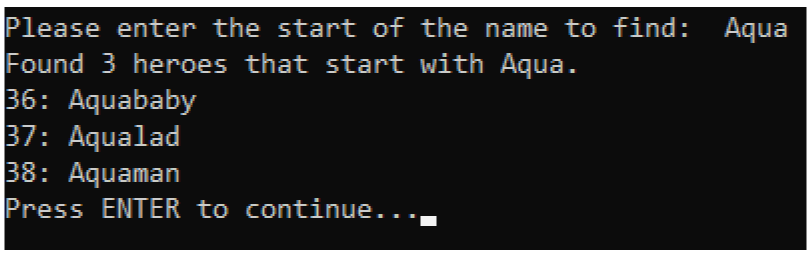
Use the **isPrefix** method of the HeroesDB class to check for a prefix. Here's an example of calling the static isPrefix method:

```
    bool prefix1 = isPrefix("Bat", "Batman");  //returns true
    bool prefix2 = isPrefix("Bat", "Aquaman"); //returns false
```

NAME	RETURNS	PARAMETERS	COMMENTS
StartsWith	vector<Hero>	string	Finds every hero whose name starts with the string parameter. Add the heroes to the vector parameter. Returns the vector.

In main (which is in **HeroesV1.cpp**), add code to case 3 of the switch. Using **Input::GetString**, ask the user to enter the first part of the names to find. Call **StartsWith** passing the string that the user enters and assign the returned vector to a vector variable. In main, print out the number of heroes found AND loop over the vector and print the Id and Name for each hero found.

For example, if the user types Aqua, the method should find all the heroes whose name starts with Aqua.



```
Please enter the start of the name to find:  Aqua
Found 3 heroes that start with Aqua.
36: Aquababy
37: Aqualad
38: Aquaman
Press ENTER to continue...
```

Make sure to commit + push after completing the section.

PART B

Lecture Videos for Part B

Day 02: Methods and Vectors Lecture:

https://fullsailedu-my.sharepoint.com/:v:/g/personal/ggirod_fullsail_com/ES_G5XbQvCdEnhGqf85AoVQBoGADuWPD2JyQZMLpcTIOoA?e=Ar9edn

Tips:

Ref parameters: C++ Functions - Pass By Reference (w3schools.com)

Removing from a vector: vector erase() and clear() in C++ - GeeksforGeeks

Part B-1: PrintHero

Add a method called **PrintHero** to the **HeroesDB** class. **Declare your method in the HeroesDB.h file and define the method in the HeroesDB.cpp file (**C++ Class Methods (w3schools.com)**).** The method should have a Hero parameter passed to it. Print the details of the Hero parameter. (see the screenshot in part B-2) **Make sure to match the formatting** (color and indention).

NAME	RETURNS	PARAMETERS	COMMENTS
PrintHero	nothing	Hero reference	Show the details of the Hero parameter.

You will call PrintHero in part B-2. That's when you'll see if it works correctly or not.

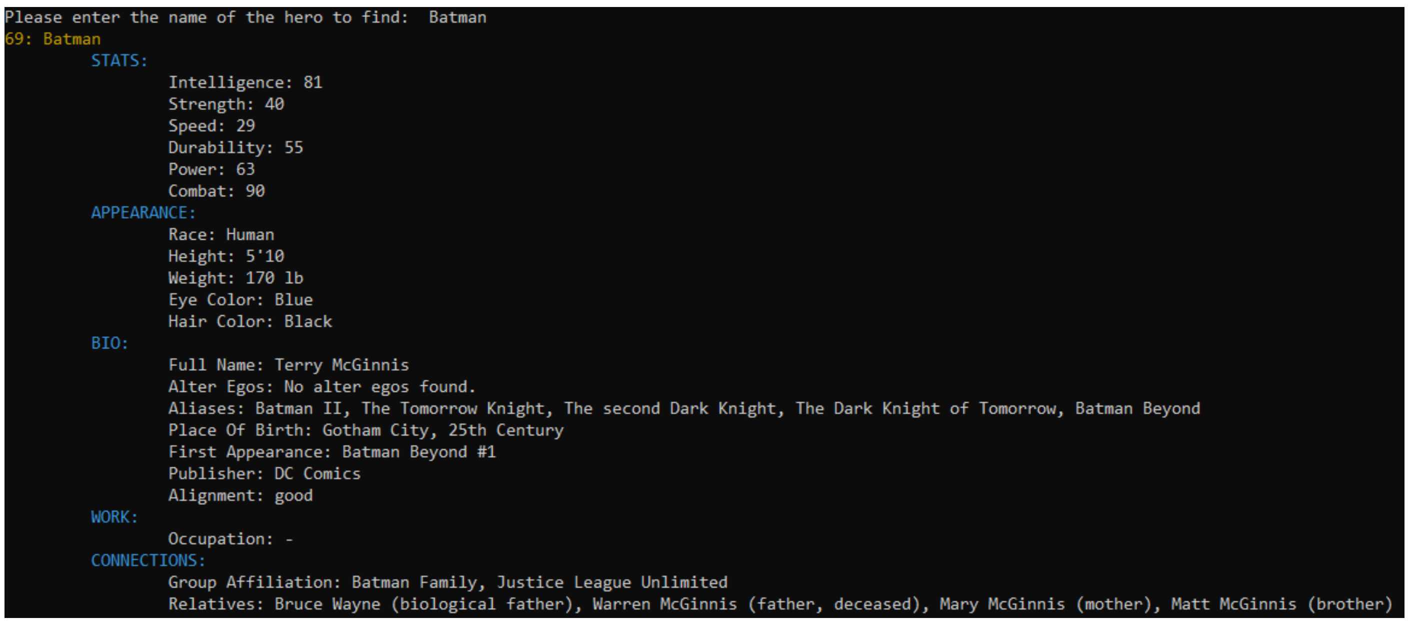
Make sure to commit + push after completing the section.

Part B-2: FindHero

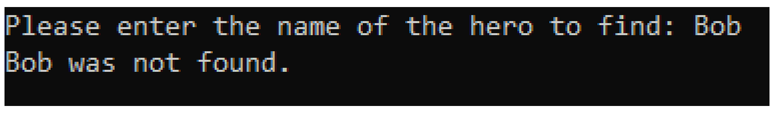
Add a method called **FindHero** to the **HeroesDB** class. **Declare your method in the HeroesDB.h file and define the method in the HeroesDB.cpp file (**C++ Class Methods (w3schools.com)**).** The method should have a string parameter for the name to search and a Hero reference parameter. The method needs to loop over the heroes vector to try to find the hero. Check if each hero name matches the parameter. If so, set the Hero parameter to the found hero, break out of the loop, and return true. When checking for a match, **make sure to ignore the case** of the name and the parameter. If the hero is not found, return false.

NAME	RETURNS	PARAMETERS	COMMENTS
FindHero	bool	String Hero reference	Returns true if a hero is found otherwise returns false. If found, the Hero ref parameter is set to the found hero.

In main (which is in **HeroesV1.cpp**), add code to case 4 of the switch. Using **Input::GetString**, ask the user to enter the name of the hero to find. Call **FindHero** passing the string that the user enters and a Hero variable. **In main**, if the returned value from FindHero is true, then call **PrintHero** passing the Hero variable else print out a message that the name was not found.



```
Please enter the name of the hero to find:  Batman
69: Batman
        STATS:
                Intelligence: 81
                Strength: 40
                Speed: 29
                Durability: 55
                Power: 63
                Combat: 90
        APPEARANCE:
                Race: Human
                Height: 5'10
                Weight: 170 lb
                Eye Color: Blue
                Hair Color: Black
        BIO:
                Full Name: Terry McGinnis
                Alter Egos: No alter egos found.
                Aliases: Batman II, The Tomorrow Knight, The second Dark Knight, The Dark Knight of Tomorrow, Batman Beyond
                Place Of Birth: Gotham City, 25th Century
                First Appearance: Batman Beyond #1
                Publisher: DC Comics
                Alignment: good
        WORK:
                Occupation: -
        CONNECTIONS:
                Group Affiliation: Batman Family, Justice League Unlimited
                Relatives: Bruce Wayne (biological father), Warren McGinnis (father, deceased), Mary McGinnis (mother), Matt McGinnis (brother)
```



```
Please enter the name of the hero to find: Bob
Bob was not found.
```

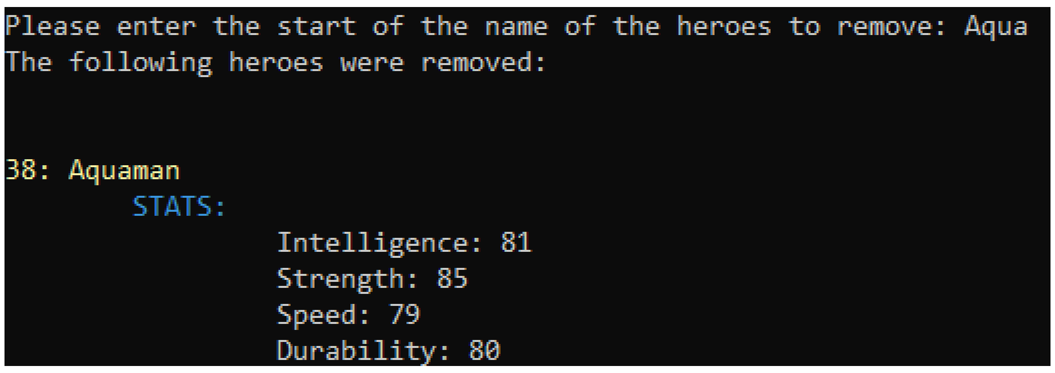
Make sure to commit + push after completing the section.

Part B-3: RemoveAllHeroes

Add a method called **RemoveAllHeroes** to the **HeroesDB** class. **Declare your method in the HeroesDB.h file and define the method in the HeroesDB.cpp file (**C++ Class Methods (w3schools.com)**).** The method should have a string parameter for the name of the hero to match and a **reference parameter** for the vector of heroes that were found and removed. Loop over the heroes vector and add every hero whose name **starts with** the string parameter to the vector parameter. Make sure to remove the hero from the heroes vector.

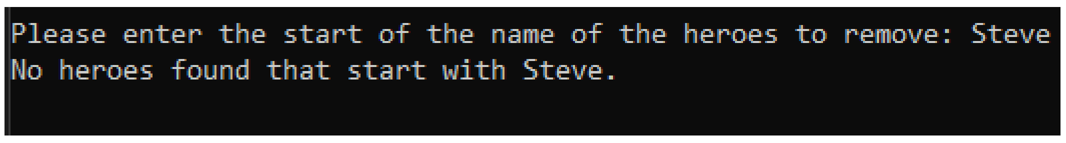
NAME	RETURNS	PARAMETERS	COMMENTS
RemoveAllHeroes	nothing	string vector<Hero> reference	Finds every hero whose name starts with the string parameter and removes the hero. Add the removed heroes to the vector parameter.

In **main** (which is in **HeroesV1.cpp**), add code to case 5 of the switch. Using **Input::GetString**, ask the user to enter the first part of the names to remove. Call **RemoveAllHeroes** passing the string that the user enters and a vector. **In main**, after calling the method, if the vector is empty, print that "No heroes found that start with <the StartsWith string the user entered>" else print "The following heroes were removed: " and loop over the vector calling **PrintHero** for each hero in the vector.



```
Please enter the start of the name of the heroes to remove: Aqua
The following heroes were removed:


38: Aquaman
        STATS:
                Intelligence: 81
                Strength: 85
                Speed: 79
                Durability: 80
```



```
Please enter the start of the name of the heroes to remove: Steve
No heroes found that start with Steve.
```

Make sure to commit + push after completing the section.

PART C

Lecture Videos for Part C

Day 03: Methods Lecture:

https://fullsailedu-my.sharepoint.com/:v:/g/personal/ggirod_fullsail_com/Eflx71MwdTFHmaHeSmyqO5UBjQGmeKlsBYhnen27VcvwJA?e=UiOgTn

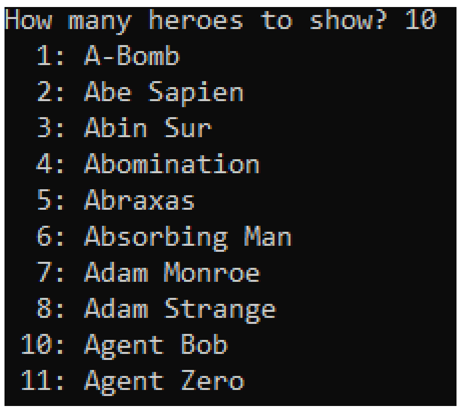
Tips:

Optional (default) parameters: C++ Functions - Default Parameter Value (Optional Parameters) (w3schools.com)

Part C-1: Optional (default) parameter

MODIFY the ShowHeroes method -- add an **optional parameter** to the ShowHeroes method. Default it to 0. In the method, if the parameter has the default value, show all the heroes else show the number of heroes that match the parameter. Example, if 10 is passed in, only show the first 10 heroes.

In **main** (which is in **HeroesV1.cpp**), add code to case 6 of the switch. Using **Input::GetInteger**, ask the user to enter the number of heroes to show. Use the **HeroesDB.Count** method to get the max value to pass to **Input::GetInteger**. Call **ShowHeroes** and pass in the number that **Input::GetInteger** returns.



```
How many heroes to show? 10
  1: A-Bomb
  2: Abe Sapien
  3: Abin Sur
  4: Abomination
  5: Abraxas
  6: Absorbing Man
  7: Adam Monroe
  8: Adam Strange
 10: Agent Bob
 11: Agent Zero
```

Make sure to commit + push after completing the section.