This coding challenge was written by Mikayla Scott under the C2 Pipeline Innovation and Curiosity Center through Wayne State University. This challenge has students work to create a MadLibs program. The challenge prompt is listed below. For more information on the STEM Challenges produced by C2 Pipeline visit https://c2pipeline.wayne.edu/stem-lab. The YouTube video that accompanies this lesson can be found at https://www.youtube.com/watch?v=11zVHHo4b6U&t=4s.

+-

"The challenge is to create a MadLibs program. The program should include multiple data types, this includes integers, strings and doubles. It should also include the use of cin and cout. When finished the program should prompt the user for all the input values, and then return the full MadLibs story back to the user!"

+-

Instructions:

1. Find an **online compiler**. The compiler that is found in the video is linked below: https://www.onlinegdb.com/online_c++_compiler
2. Once the compiler launches **delete** the pre-coded header from the .cpp file. A header is a block of comments, that tell other programmers' information about the code. The header is highlighted.

```
/******************************************************************************

                              Online C++ Compiler.
               Code, Compile, Run and Debug C++ program online.
Write your code in this editor and press "Run" button to compile and execute it.

*******************************************************************************/

#include <iostream>

using namespace std;

int main()
{
    cout<<"Hello World";

    return 0;
}
```

3. Now **define a header** by writing two back slashes **//**. The header should include, the programmers name, the date is was created, and what the program does. See completed header below. Comments do not show up in the execution of the program.

main.cpp

```
1  // This is the answer key for STEM Challenge 1
2  //+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
3  // Programmer: Mikayla Scott
4  // April 24, 2020
5  // Title: MadLibs Answer Key
6  // C2 Pipeline Innovation and Curiosity Center
7  // Wayne State University
8  //+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
9
10 #include <iostream>
```

4. Include the string library by writing **#include <string>**

main.cpp

```
1  // This is the answer key for STEM Challenge 1
2  //+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
3  // Programmer: Mikayla Scott
4  // April 24, 2020
5  // Title: MadLibs Answer Key
6  // C2 Pipeline Innovation and Curiosity Center
7  // Wayne State University
8  //+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
9
10 #include <iostream>
11 #include <string>
```

5. **Delete "Hello world**" from the old program that is in the compiler, make sure to leave the return statement, main() and the { } in the same places. Only **delete cout << "Hello world";**

```
8  //+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-
9
10 #include <iostream>
11 #include <string>
12
13 using namespace std;
14
15 int main()
16 {
17
18     return 0;
19 }
20
```

6. **Find a MadLibs template** to go off. Search google or find a template at http://www.madlibs.com/. The one that is used in the video can be found at this link: https://www.woojr.com/summer-mad-libs/funny-mad-libs/

7. Now **start declaring the variables** that the user will store their data into. There are three different variables that work for a MadLibs program, string, double and integer. For the Pizza Pizza Madlibs there will be 13 string variables and 2 integer variables. Strings store characters that are between double quotations, thigs like a sentence or word, integers store whole numbers, doubles store decimal values.

   To declare the variables follow this syntax: **VarType VarName;** VarType is the type of variable that is being declared (string, double, int,…, etc.) VarName is the name of the variable, this can be anything, it just depends on what the programmer wants to call it. See the declaration of all the variables for "Pizza Pizza" below.

   Connection to Real World: This would be the stage in the MadLibs process where a person would take into account how many blanks they would have to fill and what kinds of things go in those blanks.

   Tip: Use comments to keep track of everything!

```
13  using namespace std;
14
15  int main ()
16  {
17    //Declaring all the variables to read into
18  //adj1
19    string adj1;
20  //nationality
21    string natl;
22  //person
23    string person;
24  //noun
25    string noun1;
26  //adj2
27    string adj2;
28  //noun2
29    string noun2;
30  //adj3
31    string adj3;
32  //adj4
33    string adj4;
34  //pluralNoun
35    string pluralNoun;
36  //noun3
37    string noun3;
38  //num
39    int num;
40  //shape
41    string shape;
42  //food
43    string food;
44  //food 2
45    string food2;
46  //num2
47    int num2;
48
49    return 0;
50  }
```

8. Now that the variables have been declared, there needs to be data stored into them. The data that needs to be stored into each one of the variables, is the data that is being input by the user.

   Connection to Real World: This would be the step equivalent to asking a person all the questions associated with filling in the blanks of a MadLibs story.
   To store the data into the variables, first prompt the user by telling them what to enter: use character out (cout), followed by <<, followed by double quotations with the message inside, then << again, finished with an endl. (endl is end line, it acts as a return for the compiler, it would be the equivalent of hitting enter in a word doc on a computer.)

cout << “please enter a verb/adjective/noun” << endl;

Now, read in the information from the user.

Connection to Real World: This would be the step equivalent to the person responding to the questions that are asked by the person filling in the MadLibs.

To store the information use the following syntax: use character in (cin) followed by this token >>, then the name of the variable that needs to be read into.

cin >> VarName;

Do these steps over until you read in data for every variable that was declared.

See pictures below:

```
  //prompting the user
//reading into the variables
cout << "Please enter an adj" << endl;
cin >> adj1;

cout << "Please enter a nationality" << endl;
cin >> natl;

cout << "Please enter a person" << endl;
cin >> person;

cout << "Please enter a noun" << endl;
cin >> noun1;

cout << "Please enter an adj" << endl;
cin >> adj2;

cout << "Please enter a noun" << endl;
cin >> noun2;

cout << "Please enter an adj" << endl;
cin >> adj3;

cout << "Please enter an adj" << endl;
cin >> adj4;

cout << "Please enter a plural Noun" << endl;
cin >> pluralNoun;

cout << "Please enter a noun" << endl;
cin >> noun3;

cout << "Please enter a whole number" << endl;
cin >> num;

cout << "Please enter a shape" << endl;
cin >> shape;

cout << "Please enter a food item" << endl;
cin >> food;

cout << "Please enter another food item" << endl;
cin >> food2;

cout << "Please enter a whole number" << endl;
cin >> num2;


  return 0;
}
```

9. Now that all the data has been accounted for, the next step is to print the story back to the user.

Connection to Real World: The next step in the MadLibs process would be to read back the whole story to the person who put all the words into the story.

To do this, concatenate strings and variables to write a message. Start by using character out (cout) followed by this token << and then the first line of the madlibs story in double quotations. Write the sentence until the first blank space comes up, then close the double quotations. Use this token << and follow it by the name of the first variable

that was declared earlier in the program. Follow it by another << and continue lacing all the pieces together. Once all the variables and sentences are accounted for, finish the block of code with a << followed by an endl.

Tip: Make sure to be mindful of spaces, the compiler counts " " as a character. If the space is not there everything will run together!

See image below of the Pizza Pizza final block.

```
96  //printing madlibs to the user
97  //practicing concatenation
98  cout << "Pizza was invented by a " << adj1 << " " << natl << " chef named "
99  << person << ". To make pizza, you need to take a lump of " << noun1 <<
100 ", and make a thin, round " + adj2 + " " + noun2 + ". Then you cover it with " <<
101 adj3 << " sauce, " << adj4 << " cheese, and fresh chopped " <<
102 pluralNoun << ". Next you have to bake it in a very hot " << noun3 << ". when it is done, cut into " <<
103 num << " " << shape << ". Some kids like " << food << "pizza the best, but my favorite is the "<<
104 food2 << "pizza. If I could, I would eat pizza " << num2 << "times a day!" << endl;
105
106     return 0;
107 }
```

10. Make sure the program is finished with **return 0;** and then is closed with **}.**
11. **Run the program!**

IF YOUR PROGRAM IS THROWING ERRORS

1. Check all the semicolons throughout the program, make sure that there aren't any that are out of place.
    a. There should be semicolons after every endl**;**
    b. There should be a semicolon after every variable that is read into cin >> varName**;**
    c. There should be a semicolon after return 0**;**
    d. There should be a semicolon after every variable declaration, VarType varName**;**
2. Check the curly braces
    a. There should be a opening curly brace right after main() **{**
    b. There should be a closing curly brace right after return 0; **}**
    c. Everything after main() should be enclosed by the curly braces
3. Make sure all tokens are going the correct way
    a. << is characters/content going out to the user
    b. >> is taking characters/content in from the user
4. Make sure to check the return statement, program should be finished with **return 0;**
5. Check to make sure that cout/cin, string, int, double, endl are all **lowercase**. If they are not the compiler will not recognize them!

FOR MORE INFORMATION:

For more information please visit https://c2pipeline.wayne.edu/stem-lab. The full .cpp file can be accessed from here as well.

Happy coding!

-Mikayla