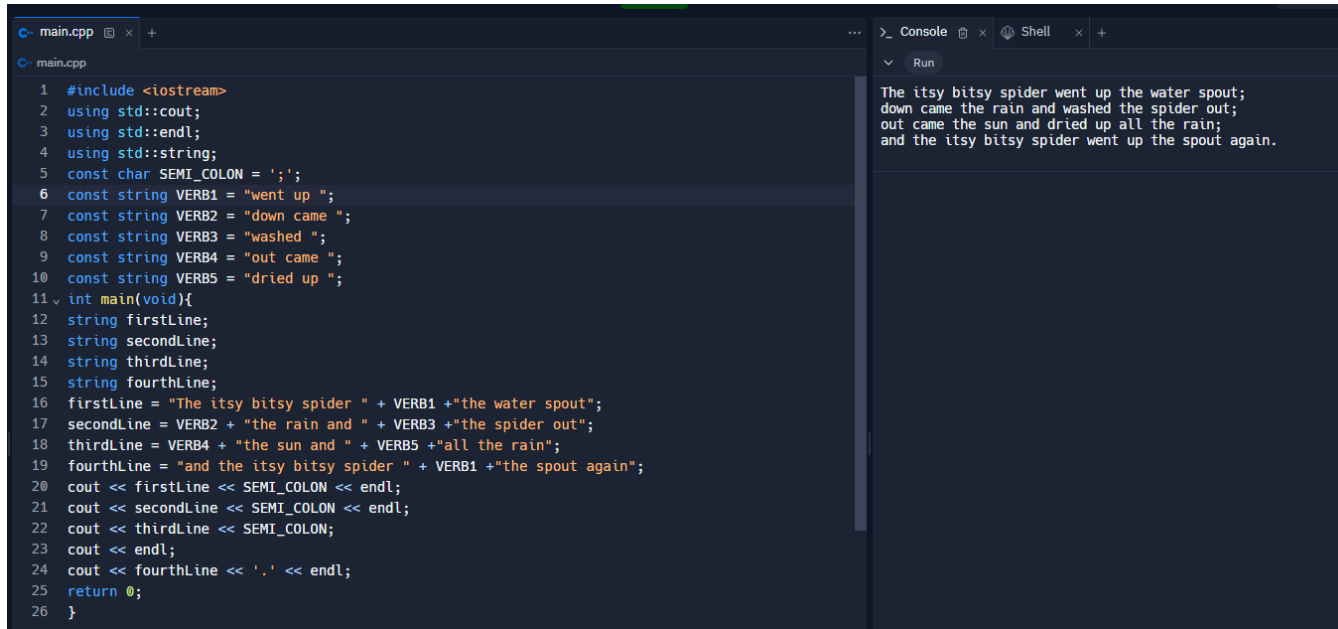


1. Let's examine / run the following C++ program regarding string data type and related operators.



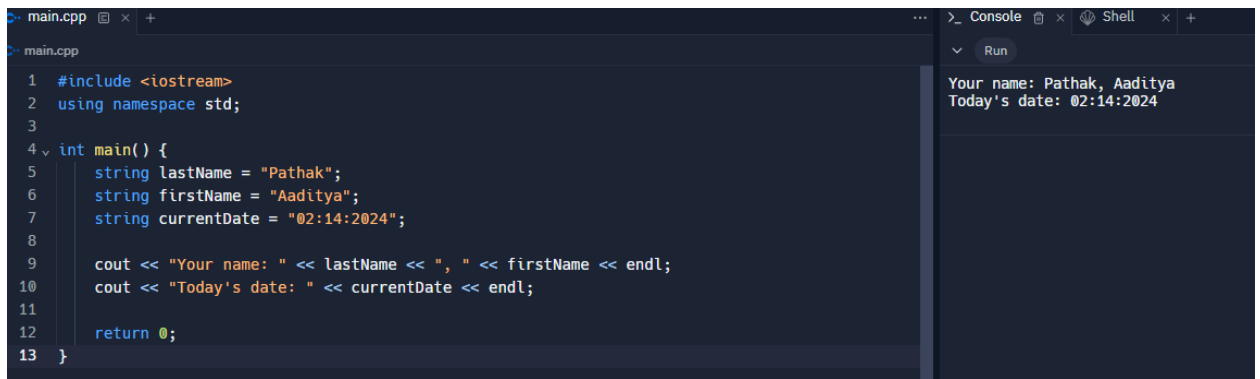
```
1 #include <iostream>
2 using std::cout;
3 using std::endl;
4 using std::string;
5 const char SEMI_COLON = ':';
6 const string VERB1 = "went up ";
7 const string VERB2 = "down came ";
8 const string VERB3 = "washed ";
9 const string VERB4 = "out came ";
10 const string VERB5 = "dried up ";
11 int main(void){
12     string firstLine;
13     string secondLine;
14     string thirdLine;
15     string fourthLine;
16     firstLine = "The itchy bitsy spider " + VERB1 + "the water spout";
17     secondLine = VERB2 + "the rain and " + VERB3 + "the spider out";
18     thirdLine = VERB4 + "the sun and " + VERB5 + "all the rain";
19     fourthLine = "and the itchy bitsy spider " + VERB1 + "the spout again";
20     cout << firstLine << SEMI_COLON << endl;
21     cout << secondLine << SEMI_COLON << endl;
22     cout << thirdLine << SEMI_COLON;
23     cout << endl;
24     cout << fourthLine << '.' << endl;
25     return 0;
26 }
```

The itsy bitsy spider went up the water spout;  
down came the rain and washed the spider out;  
out came the sun and dried up all the rain;  
and the itsy bitsy spider went up the spout again.

2. Focuses on constructing output statements. Program Shell is the outline of a program. Use this shell for Question#1 through #3

```
// Program Shell
#include <iostream>
using namespace std;
int main (){
    return 0;
}
```

- a. Question#1: Write a program to read-in from keyboard and print the following information single spaced on the screen. Use literal constants in the output statements for each of the data items to be written on the screen. Run your program to verify that the output is as specified.
  - i. your name (last name, comma, blank, first name)
  - ii. today's date (month:day:year)

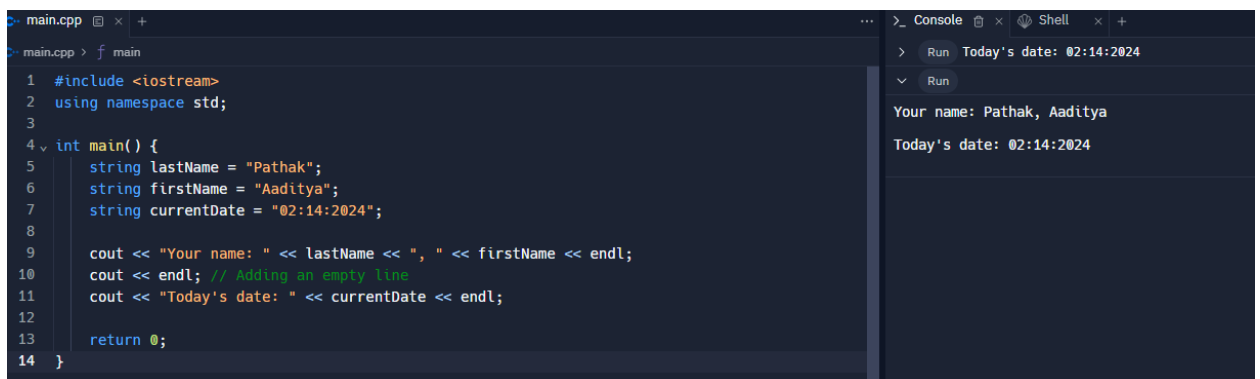


The screenshot shows a C++ IDE with a file named `main.cpp`. The code defines variables for `lastName` ("Pathak"), `firstName` ("Aaditya"), and `currentDate` ("02:14:2024"). It uses `cout` to print "Your name: " followed by `lastName`, a comma, `firstName`, and `endl`. Then it prints "Today's date: " followed by `currentDate` and `endl`. The console output shows "Your name: Pathak, Aaditya" and "Today's date: 02:14:2024" on the same line.

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     string lastName = "Pathak";
6     string firstName = "Aaditya";
7     string currentDate = "02:14:2024";
8
9     cout << "Your name: " << lastName << ", " << firstName << endl;
10    cout << "Today's date: " << currentDate << endl;
11
12    return 0;
13 }
```

Console Output:  
Your name: Pathak, Aaditya  
Today's date: 02:14:2024

**Question#2:** Change your program so that there is a space between the two lines of output.



The screenshot shows the same C++ IDE with `main.cpp`. The code is modified to add an empty line between the two `cout` statements. The console output now shows "Your name: Pathak, Aaditya" on one line and "Today's date: 02:14:2024" on the next line.

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     string lastName = "Pathak";
6     string firstName = "Aaditya";
7     string currentDate = "02:14:2024";
8
9     cout << "Your name: " << lastName << ", " << firstName << endl;
10    cout << endl; // Adding an empty line
11    cout << "Today's date: " << currentDate << endl;
12
13    return 0;
14 }
```

Console Output:  
Your name: Pathak, Aaditya  
Today's date: 02:14:2024

**Question#3:** Change your program so that your first name is printed followed by your last name, with a blank in between the names



The screenshot shows the same C++ IDE with `main.cpp`. The code is modified to print `firstName` followed by a space and `lastName` in the first `cout` statement. The console output now shows "Your name: Aaditya Pathak" on one line and "Today's date: 02:14:2024" on the next line.

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     string lastName = "Pathak";
6     string firstName = "Aaditya";
7     string currentDate = "02:14:2024";
8
9     cout << "Your name: " << firstName << " " << lastName << endl;
10    cout << endl; // Add an empty line
11    cout << "Today's date: " << currentDate << endl;
12
13    return 0;
14 }
```

Console Output:  
Your name: Aaditya Pathak  
Today's date: 02:14:2024

3. Use the following program shell for Question#1 through #3

```
// Program Strings applies string functions.
#include <iostream>
using std::cout, std::string;
int main (void){
return 0;
}
```

a. Question#1: Write a named string constant made up of your first and last names with a blank in between. Write the statements to print out the result of applying length and size to your named constant object. Compile and run your program.



The screenshot shows a C++ IDE with a file named 'main.cpp'. The code defines a string constant 'fullName' with the value 'Aaditya Pathak'. It then prints the length and size of this string. The console output shows that both the length and size are 14.

```
1 #include <iostream>
2 #include <string>
3
4 using std::cout;
5 using std::endl;
6 using std::string;
7
8 int main() {
9     const string fullName = "Aaditya Pathak";
10
11     cout << "Length of the named constant object: " << fullName.length() << endl;
12     cout << "Size of the named constant object: " << fullName.size() << endl;
13
14     return 0;
15 }
16
```

Console Output:

```
Length of the named constant object: 14
Size of the named constant object: 14
```

Question#2: Add statements to your Question#1 program to print your name formatted as last name first, followed by a comma and your first name. Use function substr to accomplish this task. Compile and run your program.



The screenshot shows the same C++ IDE with 'main.cpp'. The code is updated to extract the first and last names from 'fullName' using the 'substr' function. It then concatenates them in reverse order with a comma. The console output shows the full name and the formatted name 'Pathak, Aaditya'.

```
1 #include <iostream>
2 using std::cout;
3 using std::string;
4
5 int main() {
6     const string fullName = "Aaditya Pathak";
7
8     string lastName = fullName.substr(fullName.find(" ") + 1);
9     string firstName = fullName.substr(0, fullName.find(" "));
10
11     string formattedName = lastName + ", " + firstName;
12
13     cout << "Full name: " << fullName << endl; // Use std::endl from the std namespace
14     cout << "Formatted name: " << formattedName << endl;
15
16     return 0;
17 }
18
```

Console Output:

```
Full name: Aaditya Pathak
Formatted name: Pathak, Aaditya
```

**Question#3: Add the statements necessary to print your last name, followed by a comma and your first initial. Compile and run your program**



```
1 #include <iostream>
2 using std::cout;
3 using std::string;
4
5 int main() {
6     const string fullName = "Aaditya Pathak";
7
8     string lastName = fullName.substr(fullName.find(" ") + 1);
9     string firstName = fullName.substr(0, fullName.find(" "));
10    char firstInitial = firstName[0];
11
12    cout << "Full name: " << fullName << std::endl;
13    cout << "Last name, First initial: " << lastName << ", " << firstInitial << std::endl;
14
15    return 0;
16 }
17
```

Full name: Aaditya Pathak  
Last name, First initial: Pathak, A

**4. Use the following program shell for Question#1 through Question#4**

```
// Program Numbers sends numbers to the output stream in
// specified formats.
#include <iostream>
#include <iomanip>
using std::cout;
int main (void){
    cout << fixed << showpoint;
    return 0;
}
```

**a. Question#1: Write a program to print the following numbers right-justified in a column on the screen. Make the values named constants.**  
**1066 1492 512 1 -23**

```
main.cpp x +
main.cpp
1 #include <iostream>
2 #include <iomanip>
3 using std::cout;
4
5 int main() {
6     cout << std::fixed << std::showpoint;
7
8     const int number1 = 1066;
9     const int number2 = 1492;
10    const int number3 = 512;
11    const int number4 = 1;
12    const int number5 = -23;
13
14    cout << std::setw(6) << number1 << std::endl;
15    cout << std::setw(6) << number2 << std::endl;
16    cout << std::setw(6) << number3 << std::endl;
17    cout << std::setw(6) << number4 << std::endl;
18    cout << std::setw(6) << number5 << std::endl;
19
20    return 0;
21 }
```

Console

Run

```
1066
1492
512
1
-23
```

**Question#2:** Add two statements to your program. Calculate the floating-point result from dividing the sum of the first two values by the sum of the last three values and store it in answer. The second statement should write the contents of answer on the screen to four decimal places. (Do not forget to declare answer.)

```
main.cpp x +
main.cpp > f main
1 #include <iostream>
2 #include <iomanip>
3 using std::cout;
4
5 int main() {
6     cout << std::fixed << std::showpoint;
7
8     const int number1 = 1066;
9     const int number2 = 1492;
10    const int number3 = 512;
11    const int number4 = 1;
12    const int number5 = -23;
13
14    double answer = static_cast<double>(number1 + number2) / (number3 + number4 + number5);
15
16    cout << std::setw(6) << number1 << std::endl;
17    cout << std::setw(6) << number2 << std::endl;
18    cout << std::setw(6) << number3 << std::endl;
19    cout << std::setw(6) << number4 << std::endl;
20    cout << std::setw(6) << number5 << std::endl;
21
22    cout << "The answer is " << std::setprecision(4) << answer << std::endl;
23
24    return 0;
25 }
```


Console

Run

```
1066
1492
512
1
-23
The answer is 5.2204
```

c. Question#3: Write the following numbers right-justified in a column on the screen. Each of the data values should be written in formatted floating-point notation with two decimal places. Use field width specifications rather than listing the numbers in your program with the proper formatting. You may use either literal constants or named constants.

23.62 46.0 43.4443 100.1 98.98



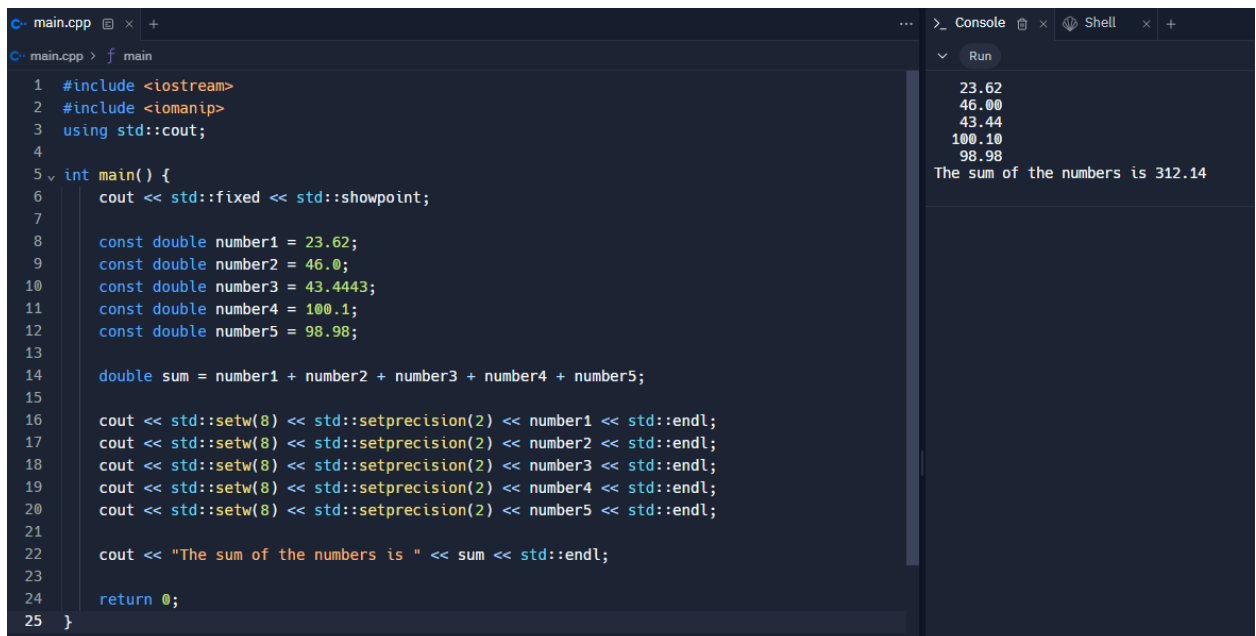
```
1 #include <iostream>
2 #include <iomanip>
3 using std::cout;
4
5 int main() {
6     cout << std::fixed << std::showpoint;
7
8     const double number1 = 23.62;
9     const double number2 = 46.0;
10    const double number3 = 43.4443;
11    const double number4 = 100.1;
12    const double number5 = 98.98;
13
14    cout << std::setw(8) << std::setprecision(2) << number1 << std::endl;
15    cout << std::setw(8) << std::setprecision(2) << number2 << std::endl;
16    cout << std::setw(8) << std::setprecision(2) << number3 << std::endl;
17    cout << std::setw(8) << std::setprecision(2) << number4 << std::endl;
18    cout << std::setw(8) << std::setprecision(2) << number5 << std::endl;
19
20    return 0;
21 }
```

Run

```
23.62
46.00
43.44
100.10
98.98
```

Question#4: Add two statements to your program for Question#3. The first statement should calculate the sum of the numbers and store the result in variable sum. The second statement should write sum on the screen, properly labeled.

The sum of the numbers is \_\_\_\_\_



```
1 #include <iostream>
2 #include <iomanip>
3 using std::cout;
4
5 int main() {
6     cout << std::fixed << std::showpoint;
7
8     const double number1 = 23.62;
9     const double number2 = 46.0;
10    const double number3 = 43.4443;
11    const double number4 = 100.1;
12    const double number5 = 98.98;
13
14    double sum = number1 + number2 + number3 + number4 + number5;
15
16    cout << std::setw(8) << std::setprecision(2) << number1 << std::endl;
17    cout << std::setw(8) << std::setprecision(2) << number2 << std::endl;
18    cout << std::setw(8) << std::setprecision(2) << number3 << std::endl;
19    cout << std::setw(8) << std::setprecision(2) << number4 << std::endl;
20    cout << std::setw(8) << std::setprecision(2) << number5 << std::endl;
21
22    cout << "The sum of the numbers is " << sum << std::endl;
23
24    return 0;
25 }
```

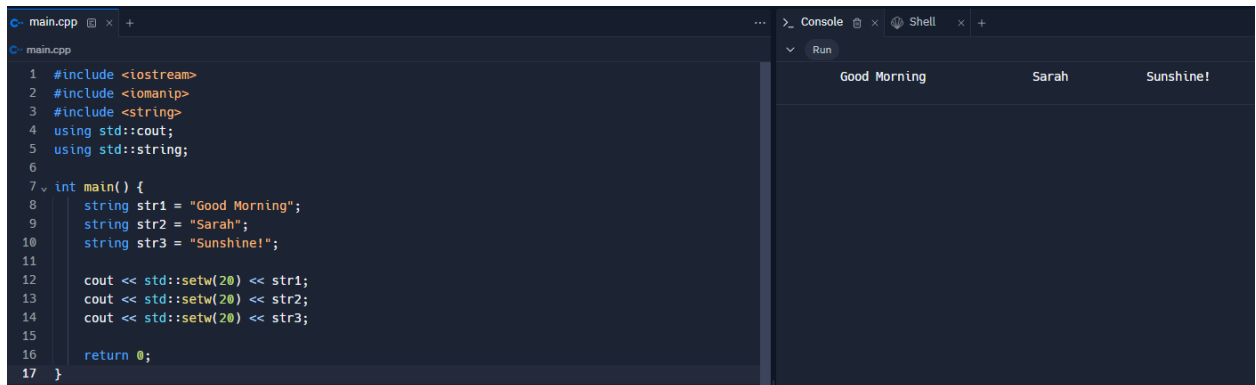
Run

```
23.62
46.00
43.44
100.10
98.98
The sum of the numbers is 312.14
```

5. Use the following program shell for Question#1through #3.

```
// Program Center sends strings to the output stream in
// specified formats.
#include <iostream>
#include <iomanip>
using std::cout;
int main (void){
return 0;
}
```

- a. Question#1: Add the statements necessary to print the following strings centered in fields of 20 characters, all on one line: "Good Morning", "Sarah", and "Sunshine!". Do not use manipulators. Compile and run your program; show your output.



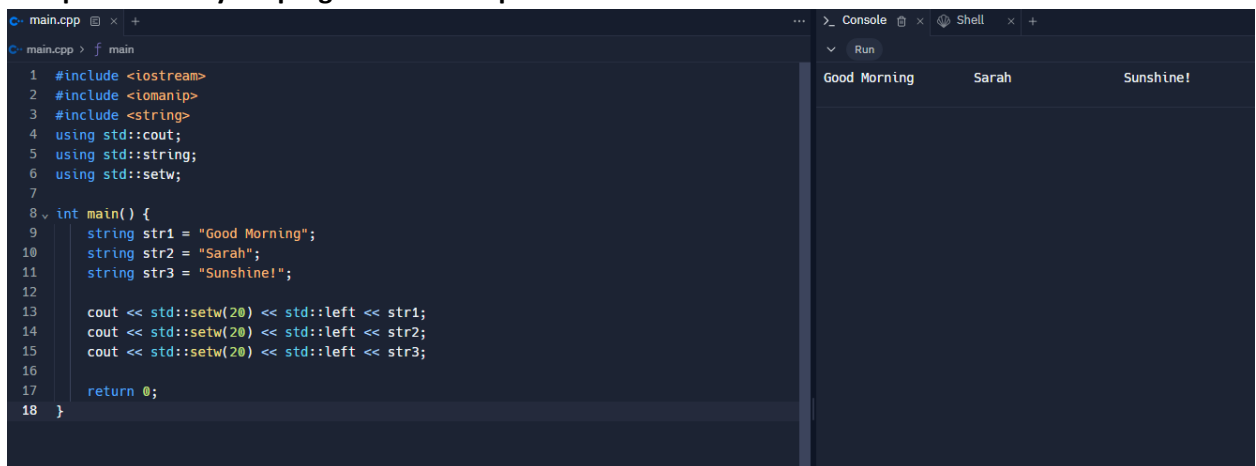
The screenshot shows a C++ IDE with a code editor on the left and a console on the right. The code in the editor is as follows:

```
1 #include <iostream>
2 #include <iomanip>
3 #include <string>
4 using std::cout;
5 using std::string;
6
7 int main() {
8     string str1 = "Good Morning";
9     string str2 = "Sarah";
10    string str3 = "Sunshine!";
11
12    cout << std::setw(20) << str1;
13    cout << std::setw(20) << str2;
14    cout << std::setw(20) << str3;
15
16    return 0;
17 }
```

The console output shows the three strings centered in 20-character fields:

```
Good Morning      Sarah      Sunshine!
```

- b. Question#2: Repeat Question#1 using manipulators to help center your strings. Compile and run your program. Your output should be the same.




The screenshot shows a C++ IDE with a code editor on the left and a console on the right. The code in the editor is as follows:

```
1 #include <iostream>
2 #include <iomanip>
3 #include <string>
4 using std::cout;
5 using std::string;
6 using std::setw;
7
8 int main() {
9     string str1 = "Good Morning";
10    string str2 = "Sarah";
11    string str3 = "Sunshine!";
12
13    cout << std::setw(20) << std::left << str1;
14    cout << std::setw(20) << std::left << str2;
15    cout << std::setw(20) << std::left << str3;
16
17    return 0;
18 }
```

The console output shows the three strings centered in 20-character fields:

```
Good Morning      Sarah      Sunshine!
```

c. Question#3: Change the program in Question#2 so that the three strings are printed on three separate lines with a blank line in between each string.



```
1 #include <iostream>
2 #include <iomanip>
3 #include <string>
4 using std::cout;
5 using std::string;
6 using std::setw;
7
8 int main() {
9     string str1 = "Good Morning";
10    string str2 = "Sarah";
11    string str3 = "Sunshine!";
12
13    cout << std::setw(20) << std::left << str1 << '\n';
14    cout << '\n';
15    cout << std::setw(20) << std::left << str2 << '\n';
16    cout << '\n';
17    cout << std::setw(20) << std::left << str3 << '\n';
18
19    return 0;
20 }
```

The screenshot shows a C++ IDE with a file named `main.cpp`. The code defines three strings: `str1 = "Good Morning"`, `str2 = "Sarah"`, and `str3 = "Sunshine!"`. Each string is printed using `cout` with `std::setw(20)` and `std::left` alignment, followed by a newline character `\n`. Additionally, a blank line is printed after each string using `cout << '\n';`. The output in the console window is:

```
Good Morning
Sarah
Sunshine!
```