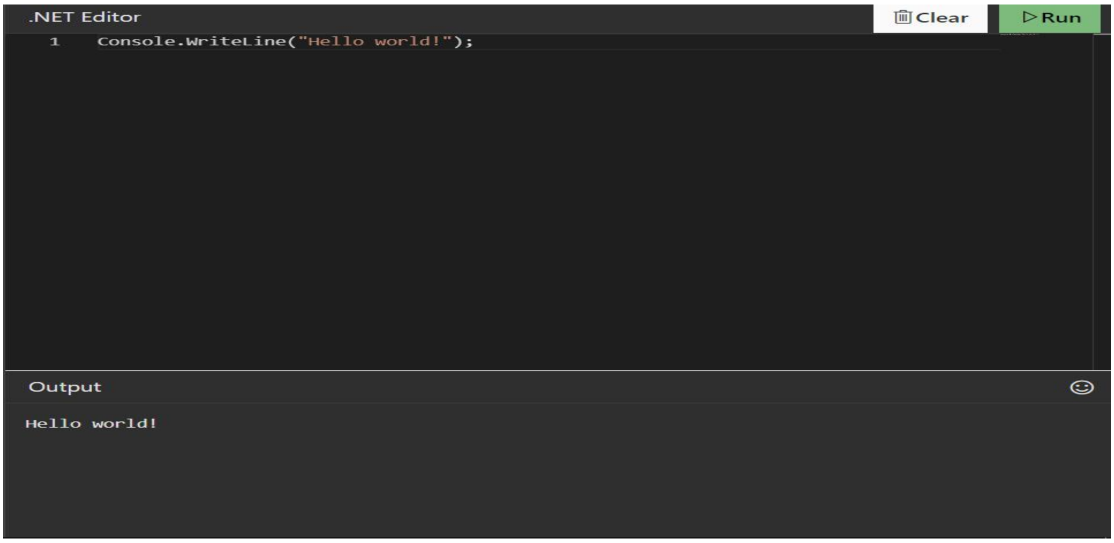
SESSION - 1:

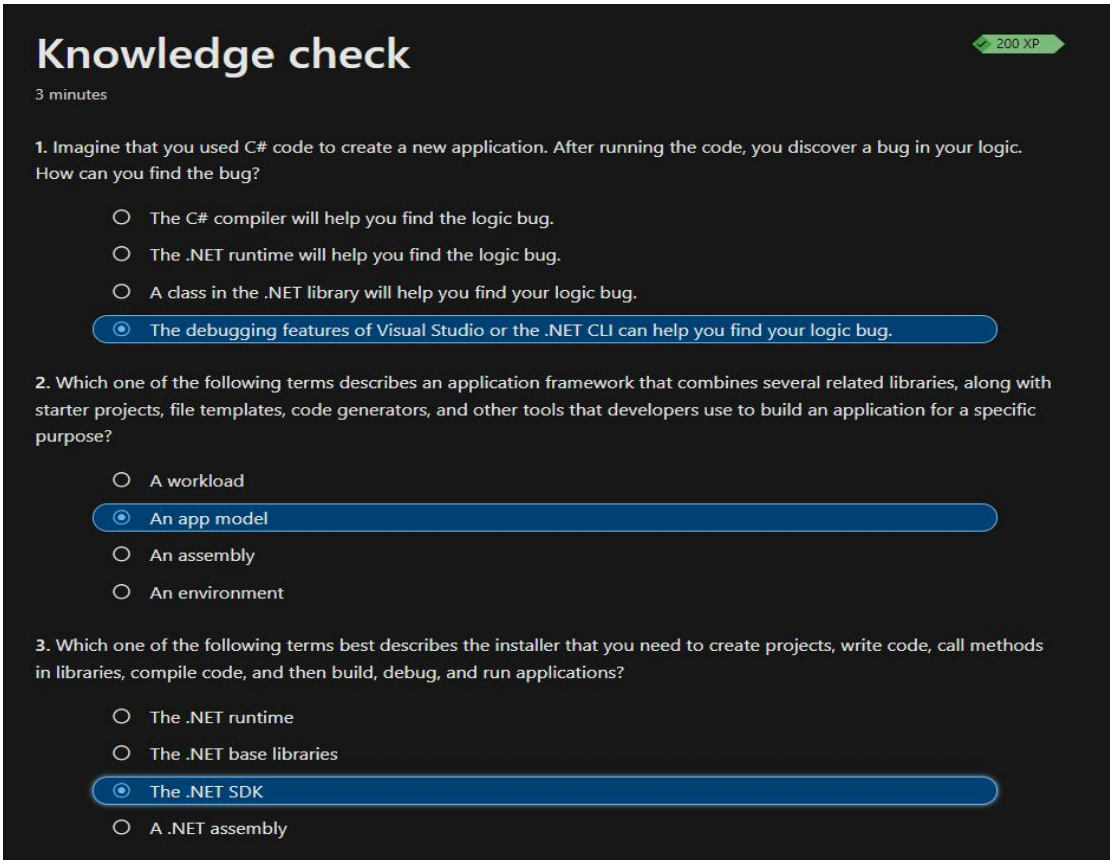
Build your first app by using Try .NET



```
Console.WriteLine("Hello world!");
```

```
Hello world!
```

MCQ :



Knowledge check

3 minutes

1. Imagine that you used C# code to create a new application. After running the code, you discover a bug in your logic. How can you find the bug?

- ○ The C# compiler will help you find the logic bug.
- ○ The .NET runtime will help you find the logic bug.
- ○ A class in the .NET library will help you find your logic bug.
- ● The debugging features of Visual Studio or the .NET CLI can help you find your logic bug.

2. Which one of the following terms describes an application framework that combines several related libraries, along with starter projects, file templates, code generators, and other tools that developers use to build an application for a specific purpose?

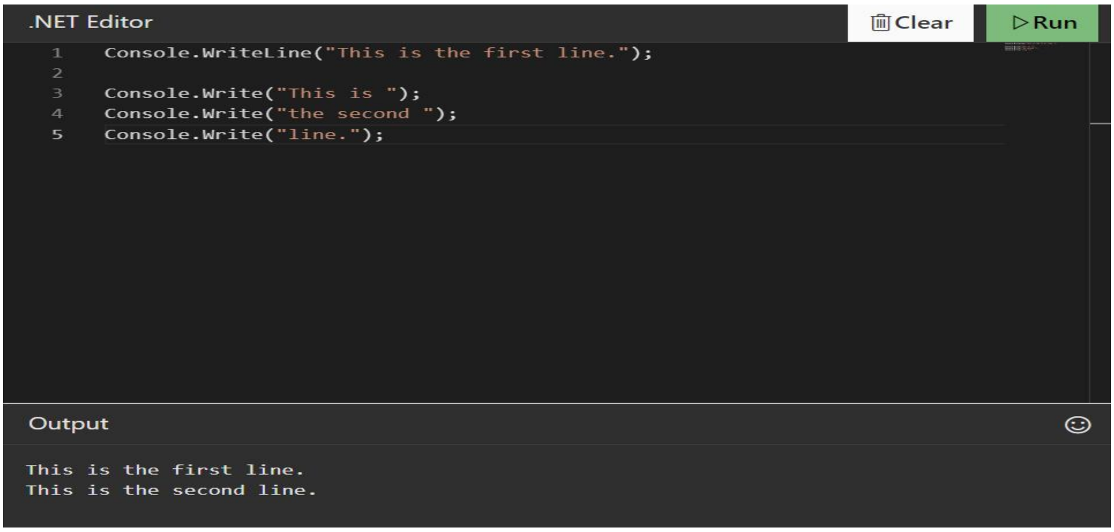
- ○ A workload
- ● An app model
- ○ An assembly
- ○ An environment

3. Which one of the following terms best describes the installer that you need to create projects, write code, call methods in libraries, compile code, and then build, debug, and run applications?

- ○ The .NET runtime
- ○ The .NET base libraries
- ● The .NET SDK
- ○ A .NET assembly

SESSION - 2:

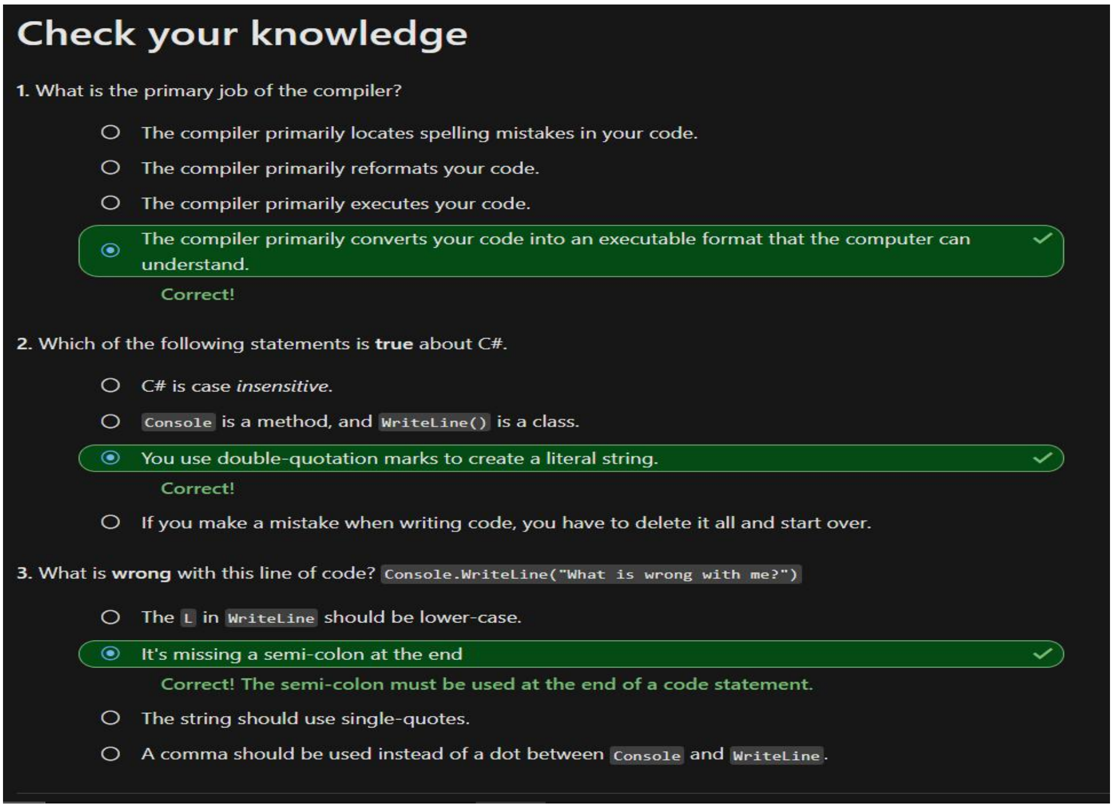
Write code to display two messages



```
Console.WriteLine("This is the first line.");

Console.Write("This is ");
Console.Write("the second ");
Console.Write("line.");
```

```
This is the first line.
This is the second line.
```

MCQ :



Check your knowledge

1. What is the primary job of the compiler?

- ○ The compiler primarily locates spelling mistakes in your code.
- ○ The compiler primarily reformats your code.
- ○ The compiler primarily executes your code.
- ◉ The compiler primarily converts your code into an executable format that the computer can understand. ✓

  Correct!

2. Which of the following statements is **true** about C#.

- ○ C# is case *insensitive*.
- ○ `Console` is a method, and `WriteLine()` is a class.
- ◉ You use double-quotation marks to create a literal string. ✓

  Correct!
- ○ If you make a mistake when writing code, you have to delete it all and start over.

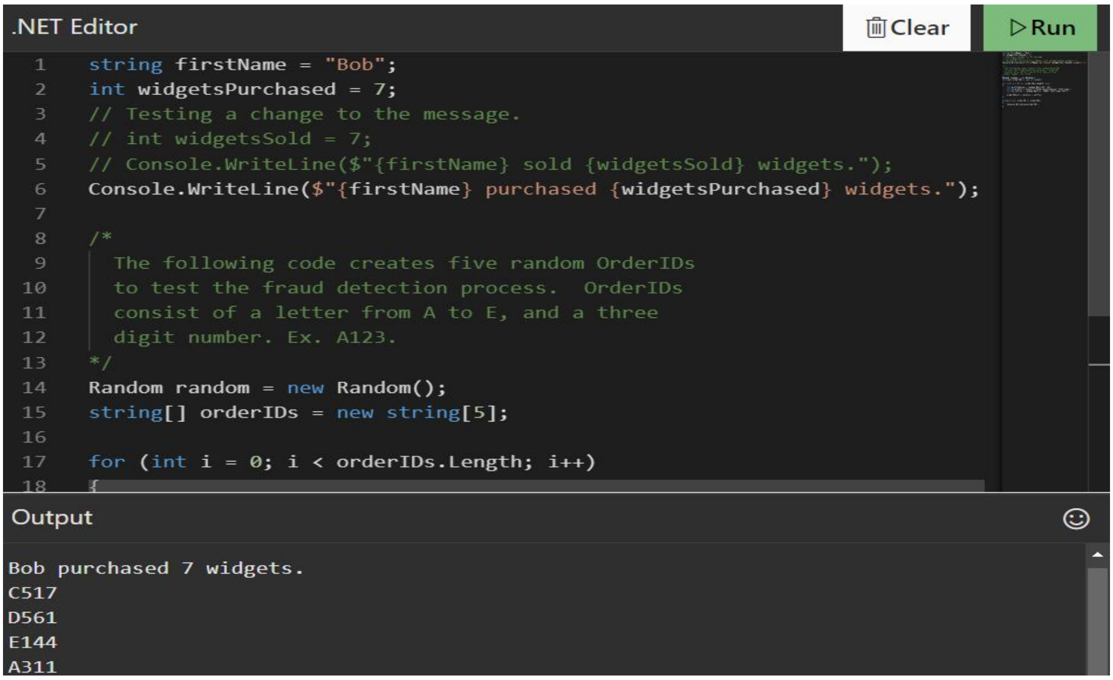
3. What is **wrong** with this line of code? `Console.WriteLine("What is wrong with me?")`

- ○ The `L` in `WriteLine` should be lower-case.
- ◉ It's missing a semi-colon at the end ✓

  Correct! The semi-colon must be used at the end of a code statement.
- ○ The string should use single-quotes.
- ○ A comma should be used instead of a dot between `Console` and `WriteLine`.

SESSION - 3:

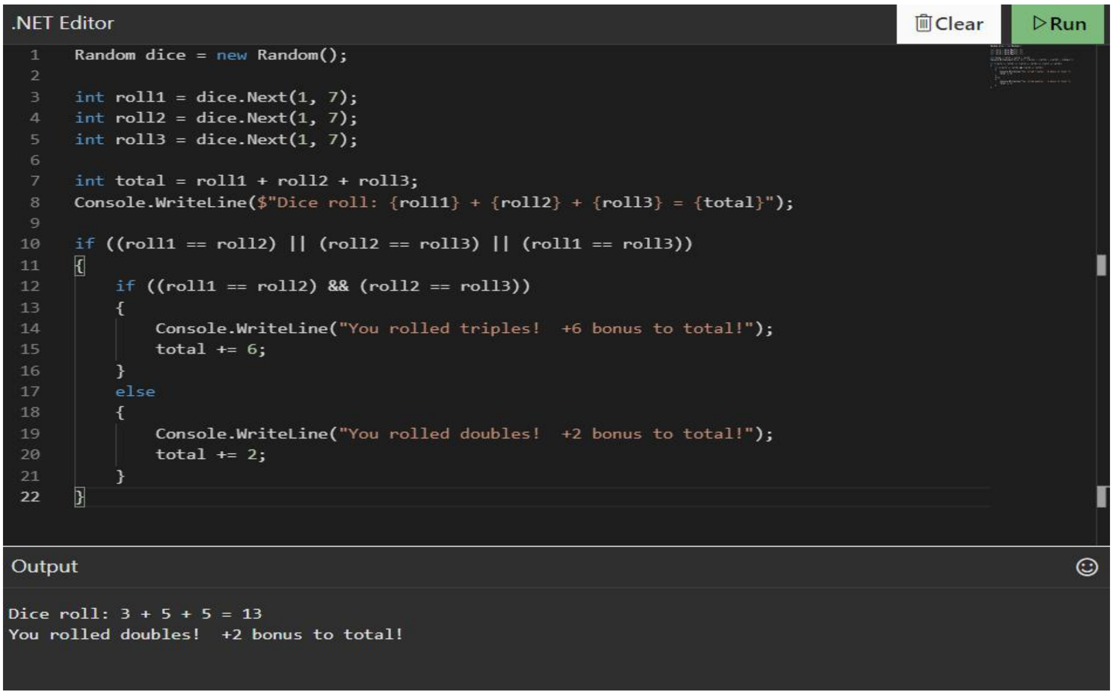
Exercise - Comment your code



```
string firstName = "Bob";
int widgetsPurchased = 7;
// Testing a change to the message.
// int widgetsSold = 7;
// Console.WriteLine($"{firstName} sold {widgetsSold} widgets.");
Console.WriteLine($"{firstName} purchased {widgetsPurchased} widgets.");

/*
  The following code creates five random OrderIDs
  to test the fraud detection process.  OrderIDs
  consist of a letter from A to E, and a three
  digit number. Ex. A123.
*/
Random random = new Random();
string[] orderIDs = new string[5];

for (int i = 0; i < orderIDs.Length; i++)
{
```

```
Bob purchased 7 widgets.
C517
D561
E144
A311
```

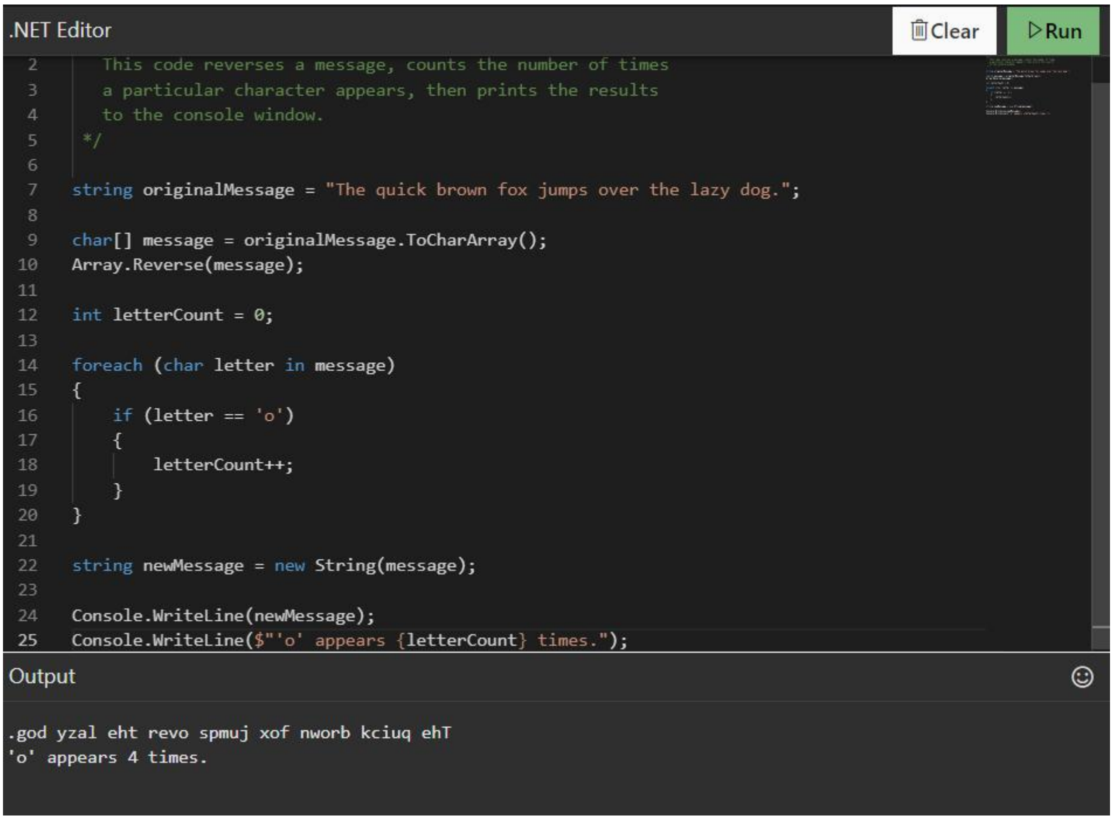
Exercise - Use whitespace



```
Random dice = new Random();

int roll1 = dice.Next(1, 7);
int roll2 = dice.Next(1, 7);
int roll3 = dice.Next(1, 7);

int total = roll1 + roll2 + roll3;
Console.WriteLine($"Dice roll: {roll1} + {roll2} + {roll3} = {total}");

if ((roll1 == roll2) || (roll2 == roll3) || (roll1 == roll3))
{
    if ((roll1 == roll2) && (roll2 == roll3))
    {
        Console.WriteLine("You rolled triples!  +6 bonus to total!");
        total += 6;
    }
    else
    {
        Console.WriteLine("You rolled doubles!  +2 bonus to total!");
        total += 2;
    }
}
```

```
Dice roll: 3 + 5 + 5 = 13
You rolled doubles!  +2 bonus to total!
```

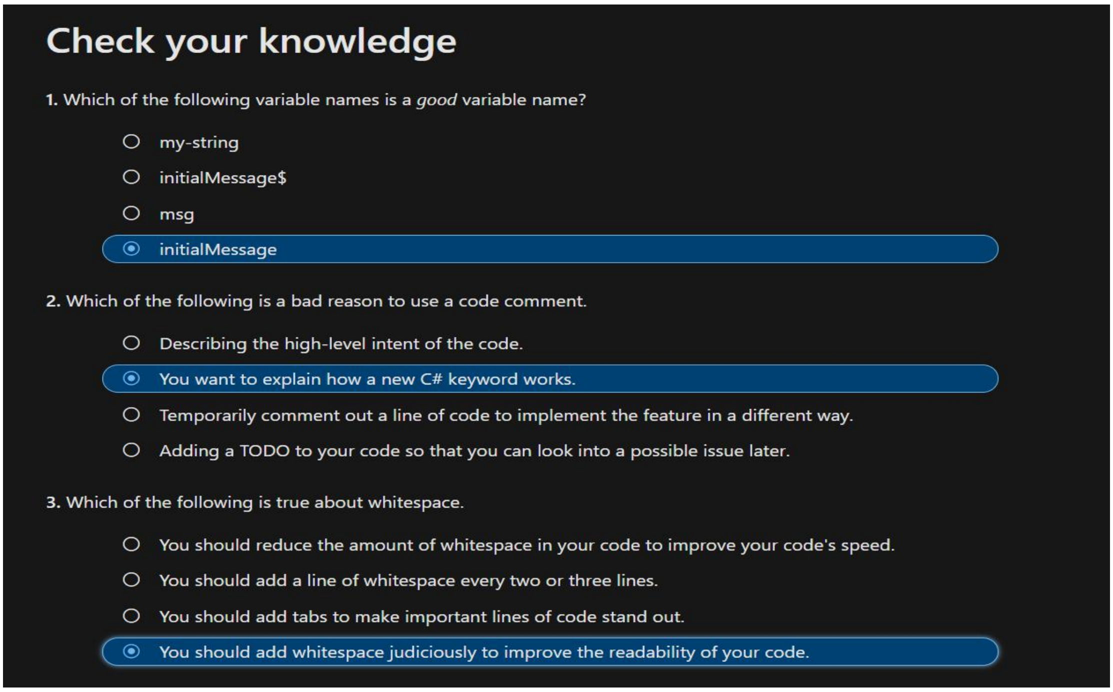
Modify the code to make it more readable



```
   This code reverses a message, counts the number of times
   a particular character appears, then prints the results
   to the console window.
 */

string originalMessage = "The quick brown fox jumps over the lazy dog.";

char[] message = originalMessage.ToCharArray();
Array.Reverse(message);

int letterCount = 0;

foreach (char letter in message)
{
    if (letter == 'o')
    {
        letterCount++;
    }
}

string newMessage = new String(message);

Console.WriteLine(newMessage);
Console.WriteLine($"'o' appears {letterCount} times.");
```

```
.god yzal eht revo spmuj xof nworb kciuq ehT
'o' appears 4 times.
```

MCQ :



Check your knowledge

1. Which of the following variable names is a *good* variable name?

○ my-string

○ initialMessage$

○ msg

● initialMessage

2. Which of the following is a bad reason to use a code comment.

○ Describing the high-level intent of the code.

● You want to explain how a new C# keyword works.

○ Temporarily comment out a line of code to implement the feature in a different way.

○ Adding a TODO to your code so that you can look into a possible issue later.

3. Which of the following is true about whitespace.

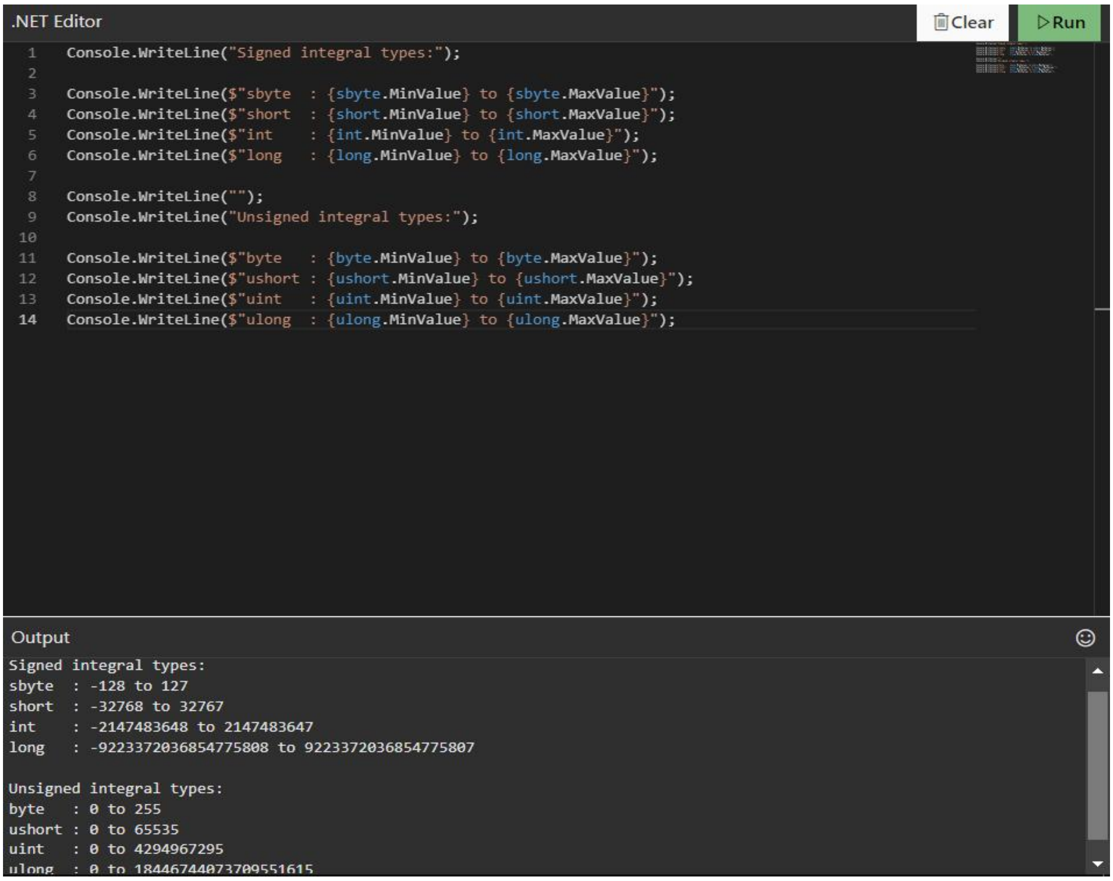
○ You should reduce the amount of whitespace in your code to improve your code's speed.

○ You should add a line of whitespace every two or three lines.

○ You should add tabs to make important lines of code stand out.

● You should add whitespace judiciously to improve the readability of your code.

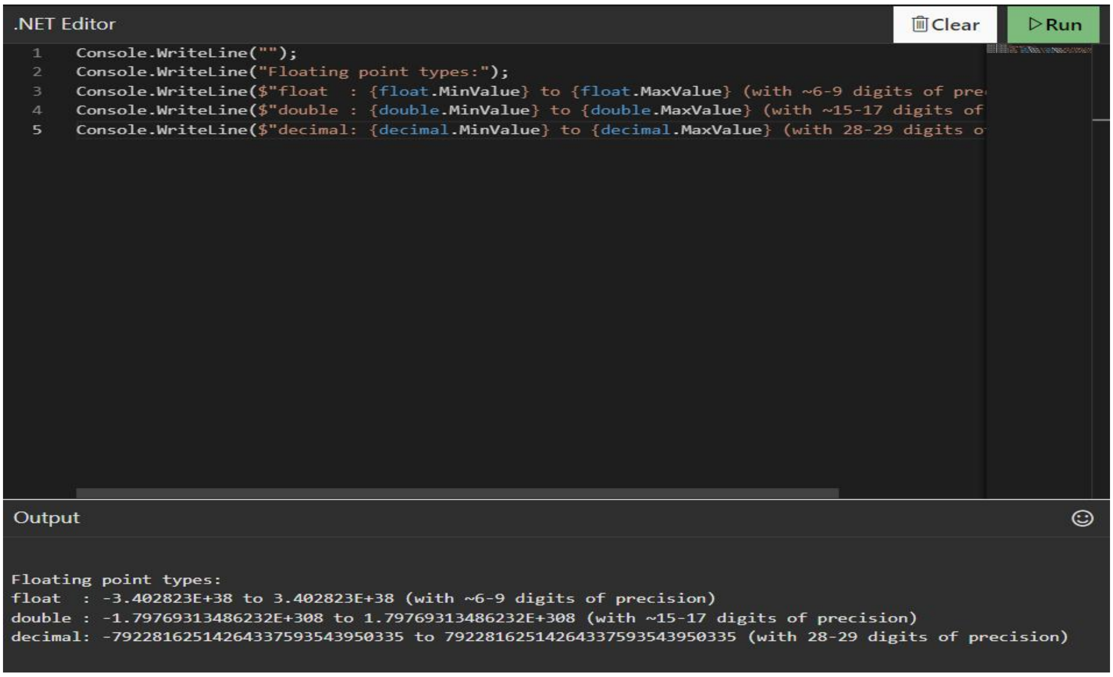
Exercise - Integral types



```
Console.WriteLine("Signed integral types:");

Console.WriteLine($"sbyte  : {sbyte.MinValue} to {sbyte.MaxValue}");
Console.WriteLine($"short  : {short.MinValue} to {short.MaxValue}");
Console.WriteLine($"int    : {int.MinValue} to {int.MaxValue}");
Console.WriteLine($"long   : {long.MinValue} to {long.MaxValue}");

Console.WriteLine("");
Console.WriteLine("Unsigned integral types:");

Console.WriteLine($"byte   : {byte.MinValue} to {byte.MaxValue}");
Console.WriteLine($"ushort : {ushort.MinValue} to {ushort.MaxValue}");
Console.WriteLine($"uint   : {uint.MinValue} to {uint.MaxValue}");
Console.WriteLine($"ulong  : {ulong.MinValue} to {ulong.MaxValue}");
```

```
Signed integral types:
sbyte  : -128 to 127
short  : -32768 to 32767
int    : -2147483648 to 2147483647
long   : -9223372036854775808 to 9223372036854775807

Unsigned integral types:
byte   : 0 to 255
ushort : 0 to 65535
uint   : 0 to 4294967295
ulong  : 0 to 18446744073709551615
```

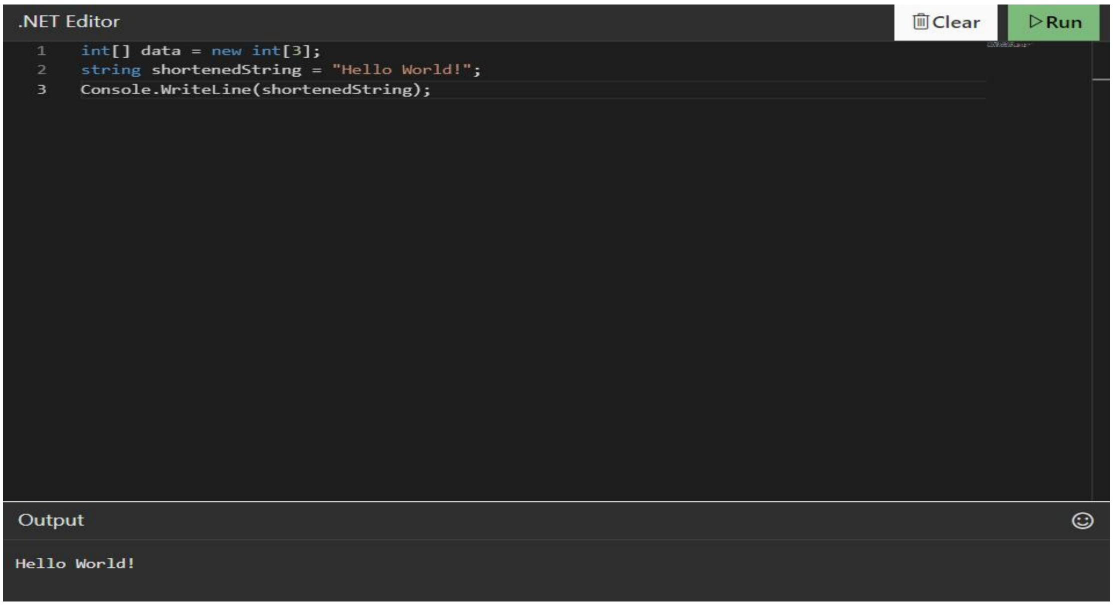
Exercise - Floating-point types



```
Console.WriteLine("");
Console.WriteLine("Floating point types:");
Console.WriteLine($"float  : {float.MinValue} to {float.MaxValue} (with ~6-9 digits of pre
Console.WriteLine($"double : {double.MinValue} to {double.MaxValue} (with ~15-17 digits of
Console.WriteLine($"decimal: {decimal.MinValue} to {decimal.MaxValue} (with 28-29 digits o
```

```

Floating point types:
float  : -3.402823E+38 to 3.402823E+38 (with ~6-9 digits of precision)
double : -1.79769313486232E+308 to 1.79769313486232E+308 (with ~15-17 digits of precision)
decimal: -79228162514264337593543950335 to 79228162514264337593543950335 (with 28-29 digits of precision)
```

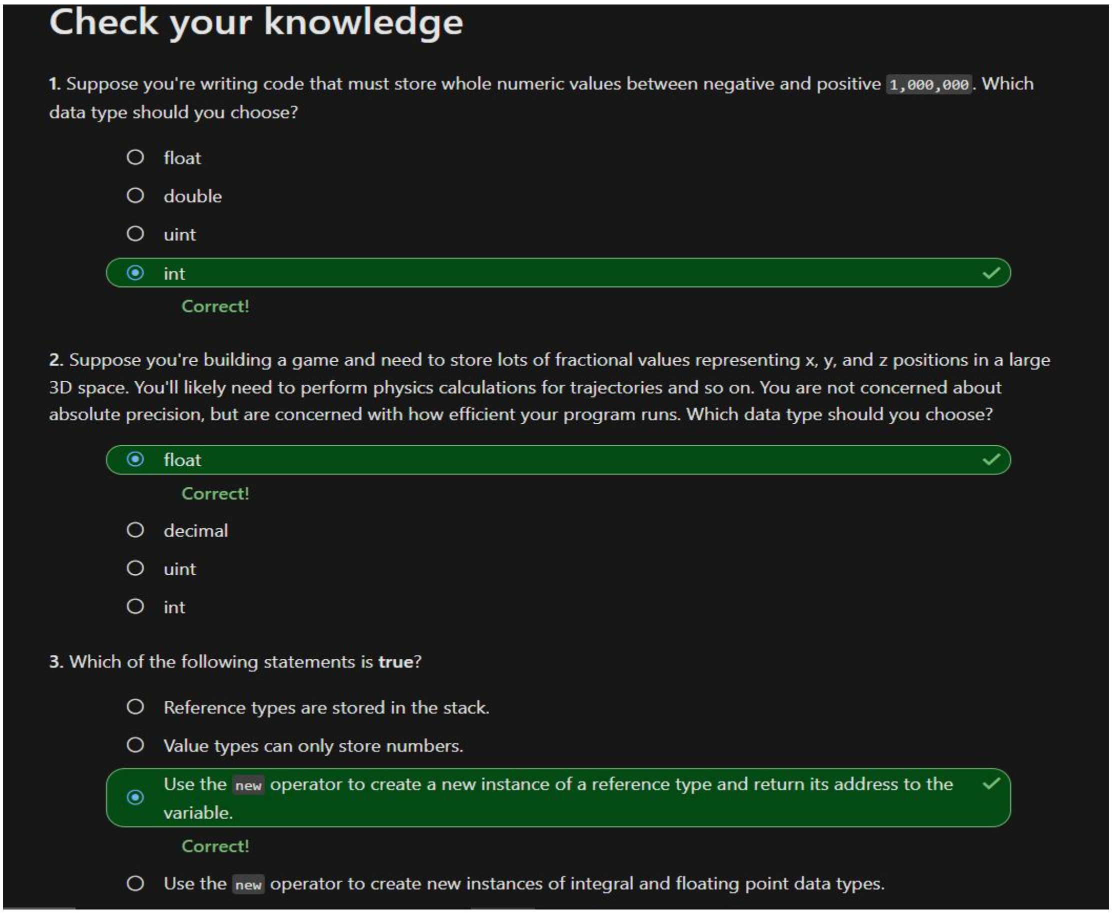
Exercise - Reference types



```
int[] data = new int[3];
string shortenedString = "Hello World!";
Console.WriteLine(shortenedString);
```

```
Hello World!
```

MCQ :



Check your knowledge

1. Suppose you're writing code that must store whole numeric values between negative and positive `1,000,000`. Which data type should you choose?

○ float

○ double

○ uint

● int ✓

Correct!

2. Suppose you're building a game and need to store lots of fractional values representing x, y, and z positions in a large 3D space. You'll likely need to perform physics calculations for trajectories and so on. You are not concerned about absolute precision, but are concerned with how efficient your program runs. Which data type should you choose?

● float ✓

Correct!

○ decimal

○ uint

○ int

3. Which of the following statements is **true**?

○ Reference types are stored in the stack.

○ Value types can only store numbers.

● Use the `new` operator to create a new instance of a reference type and return its address to the variable. ✓

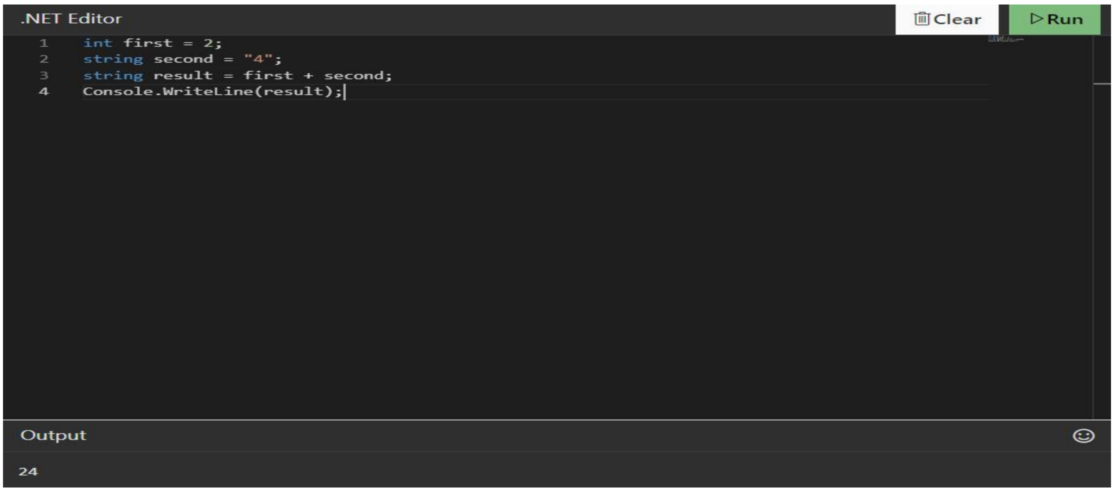
Correct!

○ Use the `new` operator to create new instances of integral and floating point data types.

SESSION -4 :

Exercise - data type casting and conversion

Question 1: Is it possible, depending on the value, that attempting to change the value's data type would throw an exception at run time?

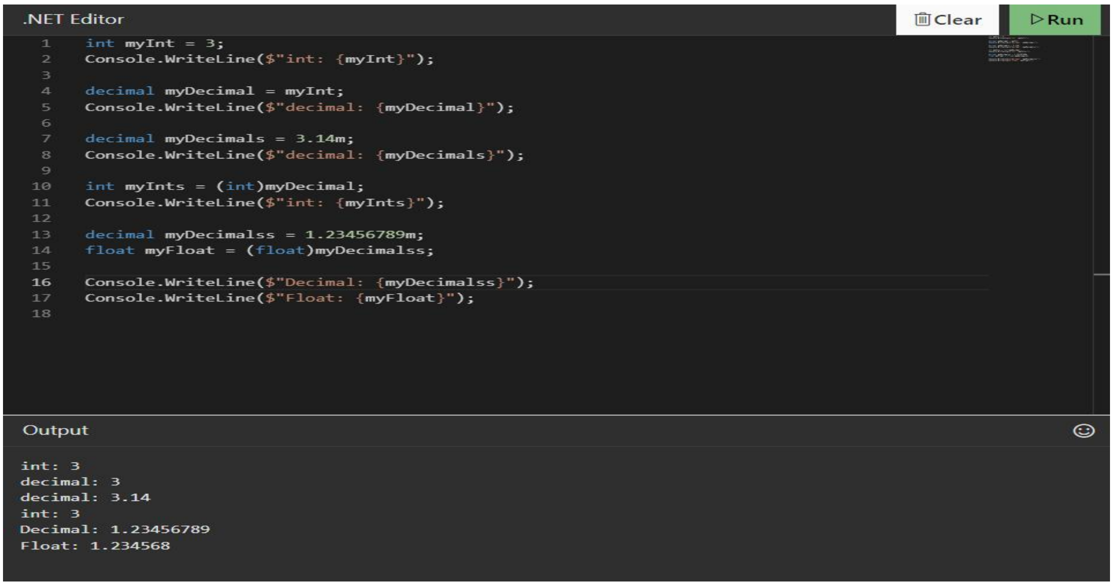


```
int first = 2;
string second = "4";
string result = first + second;
Console.WriteLine(result);
```

```
24
```

Question 2: Is it possible, depending on the value, that attempting to change the value's data type would result in a loss of information?

Modify the code to perform an implicit conversion

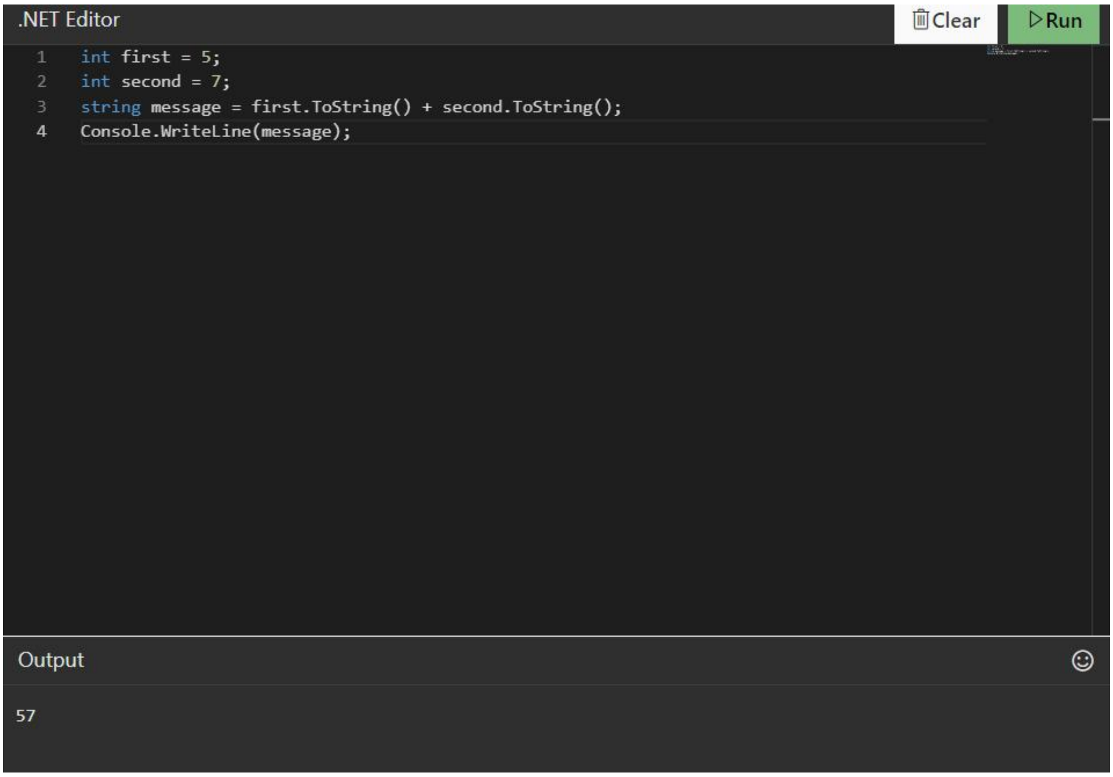


```
int myInt = 3;
Console.WriteLine($"int: {myInt}");

decimal myDecimal = myInt;
Console.WriteLine($"decimal: {myDecimal}");

decimal myDecimals = 3.14m;
Console.WriteLine($"decimal: {myDecimals}");

int myInts = (int)myDecimal;
Console.WriteLine($"int: {myInts}");

decimal myDecimalss = 1.23456789m;
float myFloat = (float)myDecimalss;

Console.WriteLine($"Decimal: {myDecimalss}");
Console.WriteLine($"Float: {myFloat}");
```

```
int: 3
decimal: 3
decimal: 3.14
int: 3
Decimal: 1.23456789
Float: 1.234568
```

Performing Data Conversions

Modify the code to convert a number to a string using the ToString() helper method

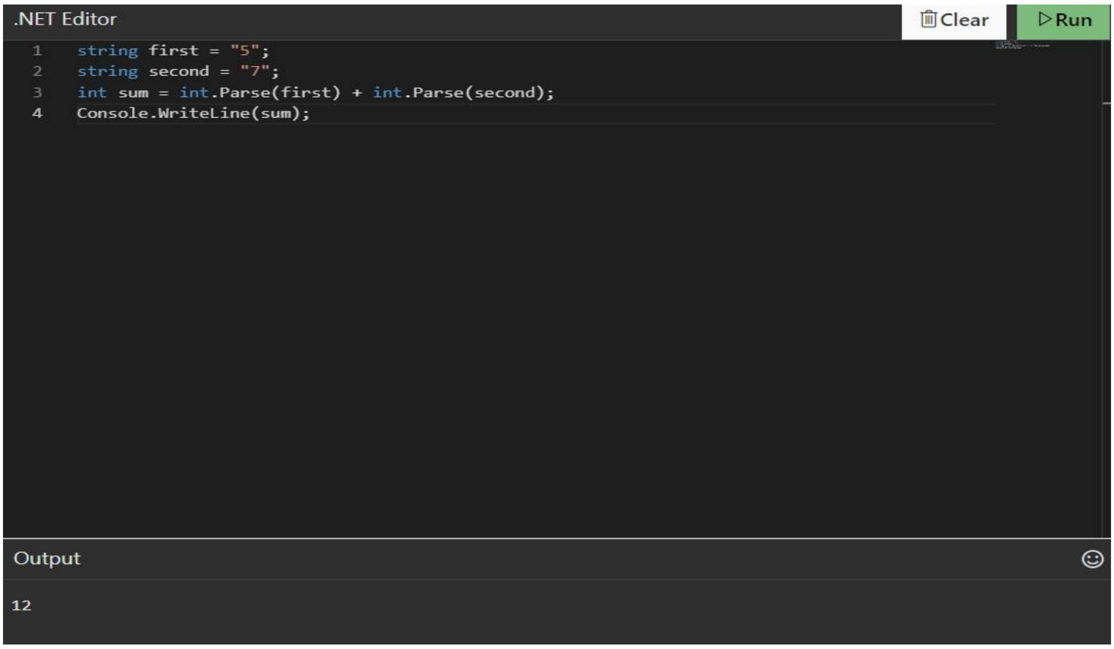


```
int first = 5;
int second = 7;
string message = first.ToString() + second.ToString();
Console.WriteLine(message);
```

```
57
```

Explicitly converting a string to a number

Modify the code to convert a string to an int using the Parse() helper method

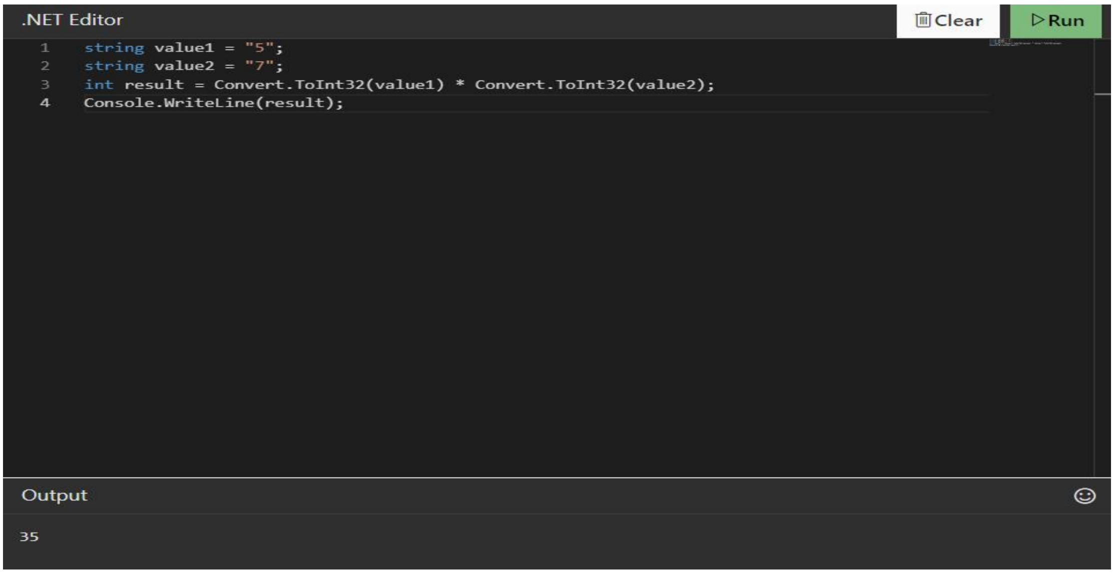


```
string first = "5";
string second = "7";
int sum = int.Parse(first) + int.Parse(second);
Console.WriteLine(sum);
```

```
12
```

Data Conversion using the Convert class

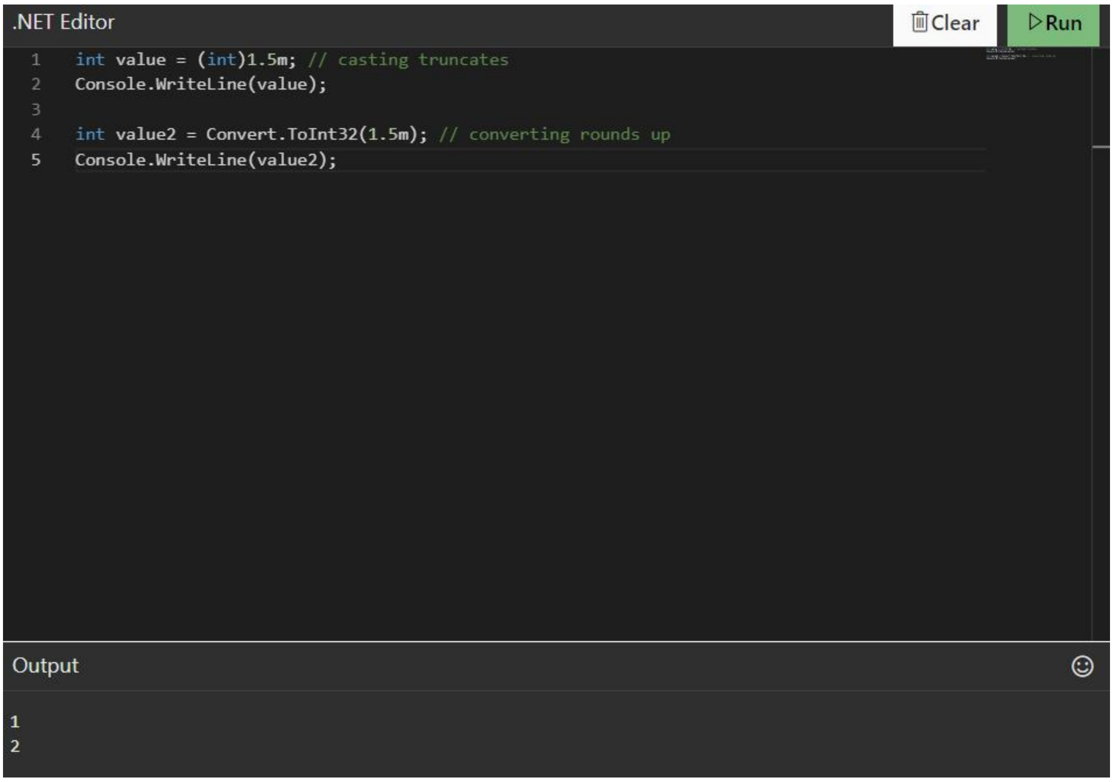
Modify the code to convert a string to a number using the Convert class



```
string value1 = "5";
string value2 = "7";
int result = Convert.ToInt32(value1) * Convert.ToInt32(value2);
Console.WriteLine(result);
```

```
35
```

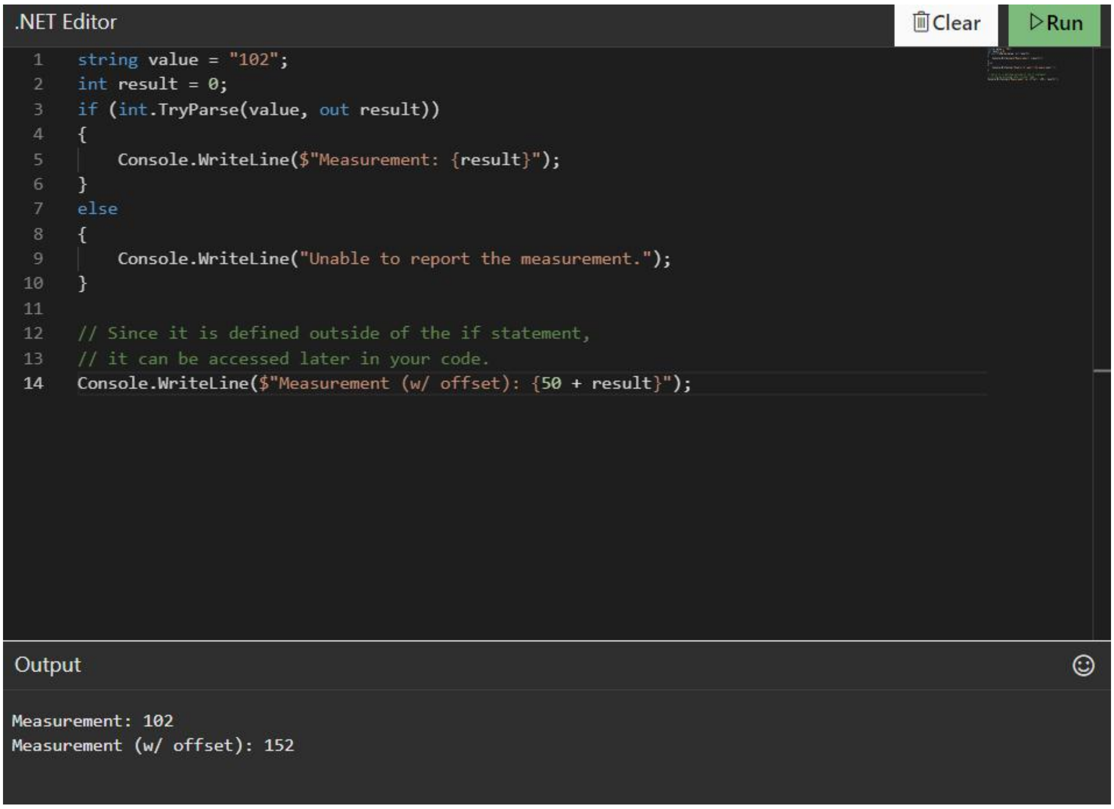
Modify the code to compare casting and converting a decimal into an int



```
int value = (int)1.5m; // casting truncates
Console.WriteLine(value);

int value2 = Convert.ToInt32(1.5m); // converting rounds up
Console.WriteLine(value2);
```

```
1
2
```

Exercise - the TryParse() method

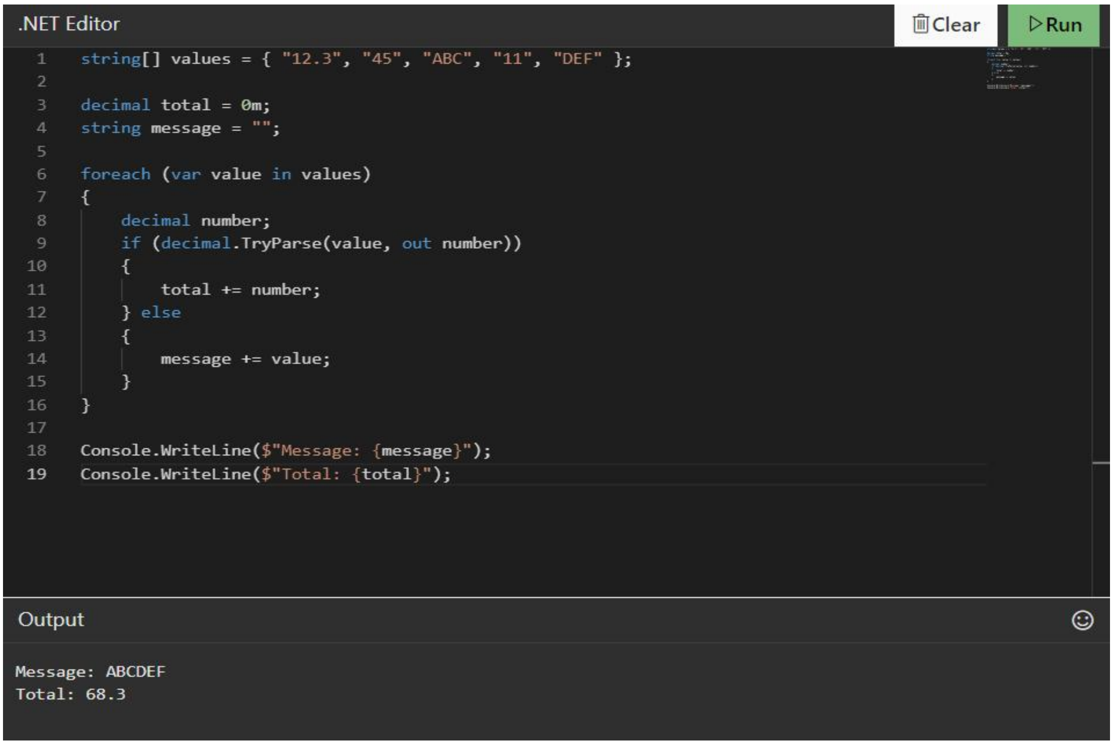


```csharp
string value = "102";
int result = 0;
if (int.TryParse(value, out result))
{
    Console.WriteLine($"Measurement: {result}");
}
else
{
    Console.WriteLine("Unable to report the measurement.");
}

// Since it is defined outside of the if statement,
// it can be accessed later in your code.
Console.WriteLine($"Measurement (w/ offset): {50 + result}");
```

```
Measurement: 102
Measurement (w/ offset): 152
```

First challenge

This module will feature two code challenges. This first challenge forces you to split up the data depending on its type and either concatenate or add the data accordingly.

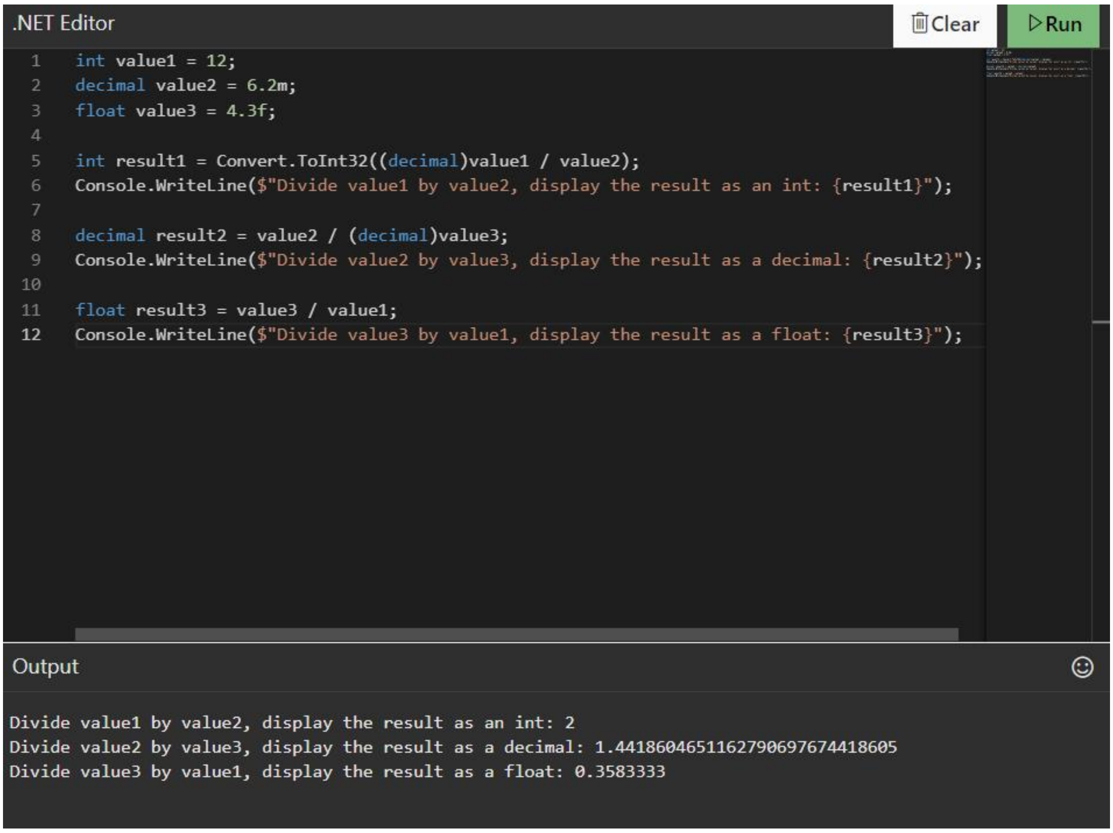


```csharp
string[] values = { "12.3", "45", "ABC", "11", "DEF" };

decimal total = 0m;
string message = "";

foreach (var value in values)
{
    decimal number;
    if (decimal.TryParse(value, out number))
    {
        total += number;
    } else
    {
        message += value;
    }
}

Console.WriteLine($"Message: {message}");
Console.WriteLine($"Total: {total}");
```

```
Message: ABCDEF
Total: 68.3
```

Second challenge

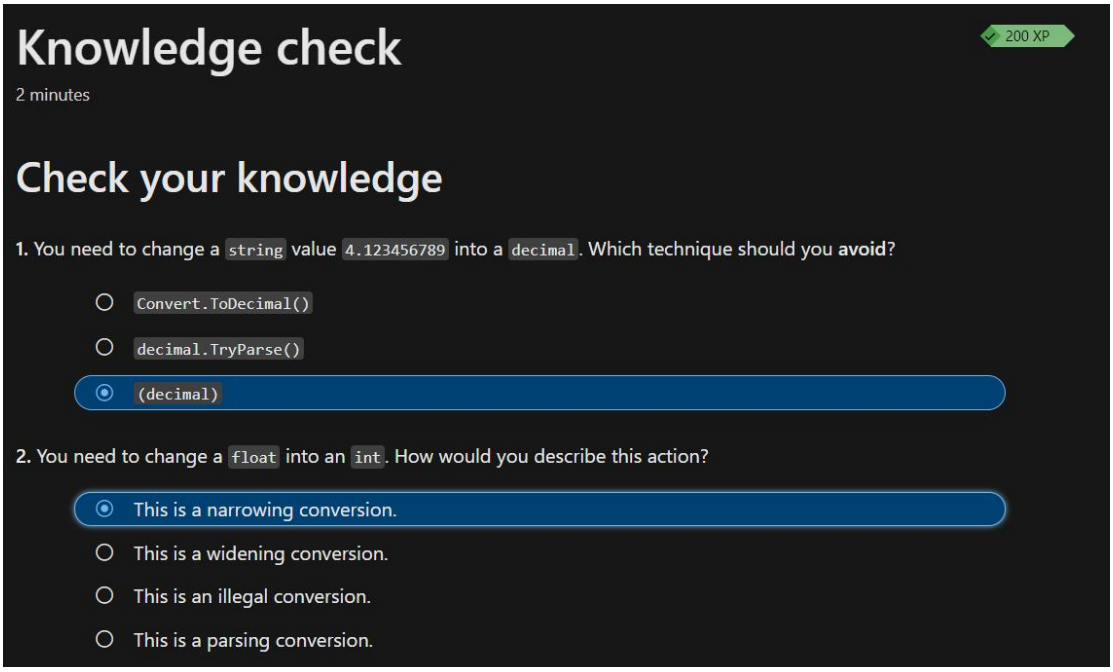
The following challenge will force you to understand the implications of casting values considering the impact of narrowing and widening conversions.



```csharp
int value1 = 12;
decimal value2 = 6.2m;
float value3 = 4.3f;

int result1 = Convert.ToInt32((decimal)value1 / value2);
Console.WriteLine($"Divide value1 by value2, display the result as an int: {result1}");

decimal result2 = value2 / (decimal)value3;
Console.WriteLine($"Divide value2 by value3, display the result as a decimal: {result2}");

float result3 = value3 / value1;
Console.WriteLine($"Divide value3 by value1, display the result as a float: {result3}");
```

```
Divide value1 by value2, display the result as an int: 2
Divide value2 by value3, display the result as a decimal: 1.4418604651162790697674418605
Divide value3 by value1, display the result as a float: 0.3583333
```

MCQ :



Knowledge check

2 minutes

Check your knowledge

1. You need to change a `string` value `4.123456789` into a `decimal`. Which technique should you **avoid**?

○ `Convert.ToDecimal()`

○ `decimal.TryParse()`

● `(decimal)`

2. You need to change a `float` into an `int`. How would you describe this action?

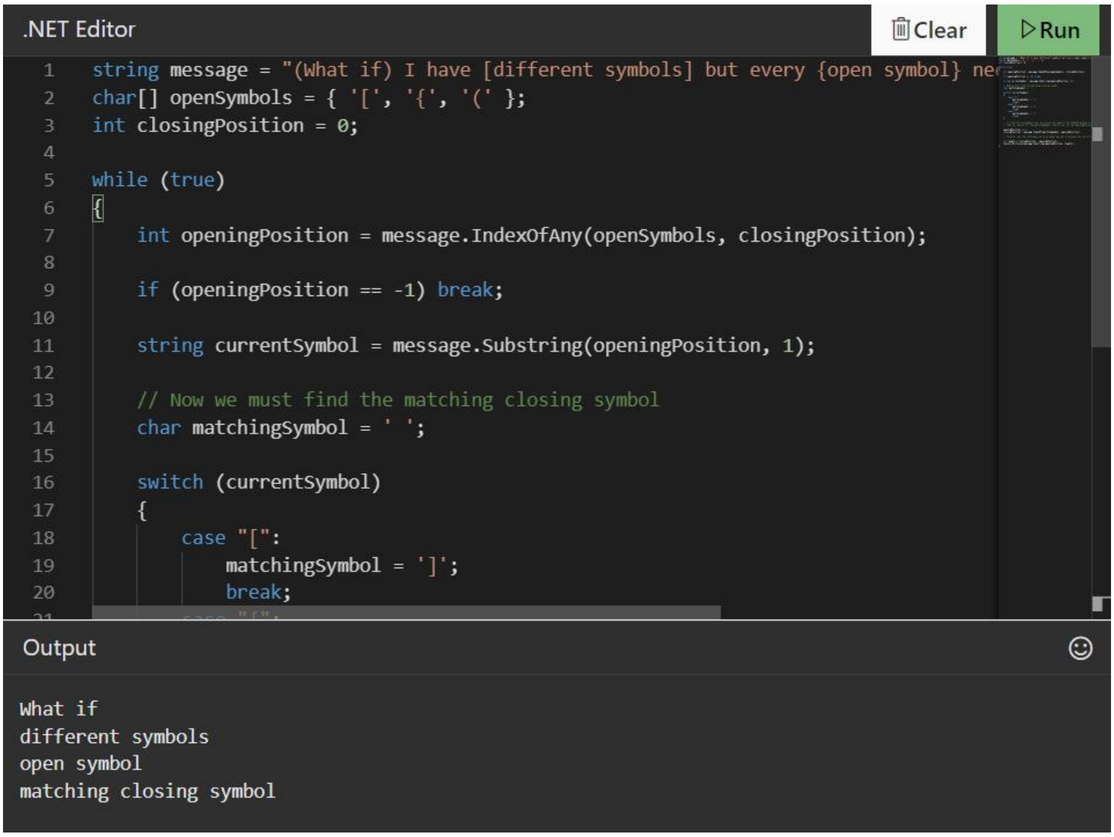
● This is a narrowing conversion.

○ This is a widening conversion.

○ This is an illegal conversion.

○ This is a parsing conversion.

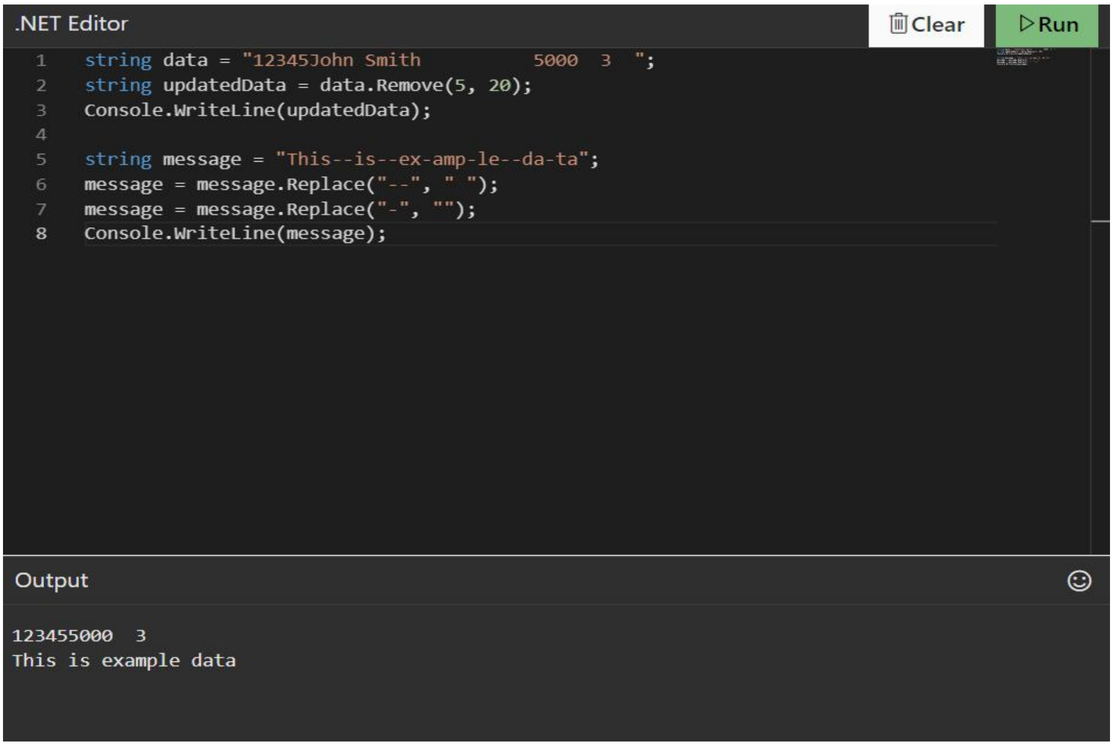
Exercise - Use the string's IndexOf() and Substring() helper methods



```
string message = "(What if) I have [different symbols] but every {open symbol} ne
char[] openSymbols = { '[', '{', '(' };
int closingPosition = 0;

while (true)
{
    int openingPosition = message.IndexOfAny(openSymbols, closingPosition);

    if (openingPosition == -1) break;

    string currentSymbol = message.Substring(openingPosition, 1);

    // Now we must find the matching closing symbol
    char matchingSymbol = ' ';

    switch (currentSymbol)
    {
        case "[":
            matchingSymbol = ']';
            break;
```

```
What if
different symbols
open symbol
matching closing symbol
```

Exercise - Use the Remove() and Replace() methods

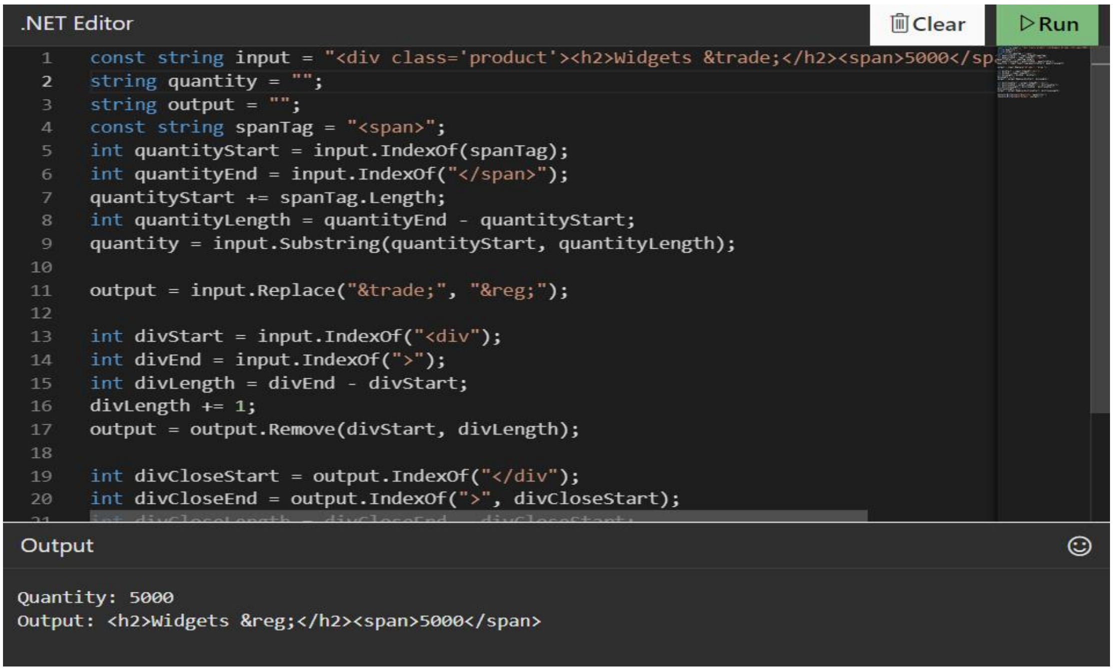


```
string data = "12345John Smith          5000  3  ";
string updatedData = data.Remove(5, 20);
Console.WriteLine(updatedData);

string message = "This--is--ex-amp-le--da-ta";
message = message.Replace("--", " ");
message = message.Replace("-", "");
Console.WriteLine(message);
```

```
123455000  3
This is example data
```

Challenge

In this challenge, you'll work with a string that contains a fragment of HTML. You'll extract data from the HTML fragment, replace some of its content, and remove other parts of its content to achieve the desired output.

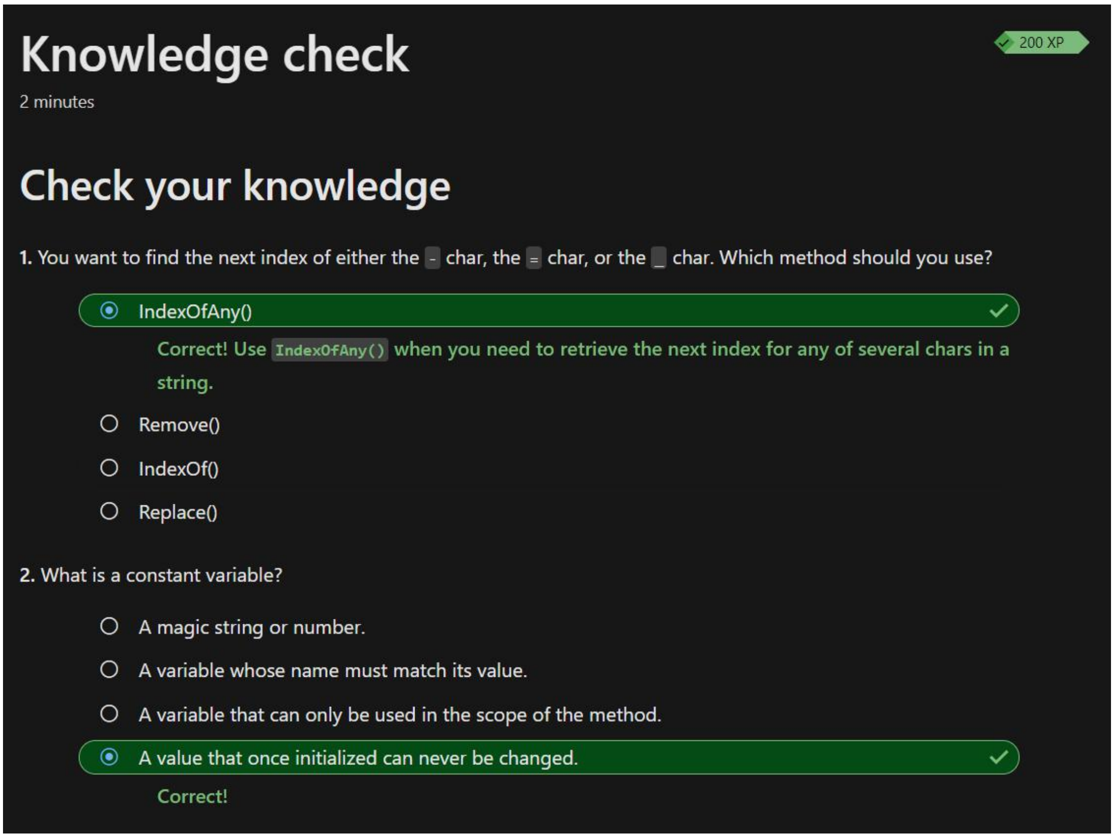


```
const string input = "<div class='product'><h2>Widgets &trade;</h2><span>5000</sp
string quantity = "";
string output = "";
const string spanTag = "<span>";
int quantityStart = input.IndexOf(spanTag);
int quantityEnd = input.IndexOf("</span>");
quantityStart += spanTag.Length;
int quantityLength = quantityEnd - quantityStart;
quantity = input.Substring(quantityStart, quantityLength);

output = input.Replace("&trade;", "&reg;");

int divStart = input.IndexOf("<div");
int divEnd = input.IndexOf(">");
int divLength = divEnd - divStart;
divLength += 1;
output = output.Remove(divStart, divLength);

int divCloseStart = output.IndexOf("</div");
int divCloseEnd = output.IndexOf(">", divCloseStart);
int divCloseLength = divCloseEnd - divCloseStart;
```

Output

```
Quantity: 5000
Output: <h2>Widgets &reg;</h2><span>5000</span>
```

MCQ :



Knowledge check

2 minutes

Check your knowledge

1. You want to find the next index of either the `-` char, the `=` char, or the `_` char. Which method should you use?

● IndexOfAny() ✓

Correct! Use `IndexOfAny()` when you need to retrieve the next index for any of several chars in a string.

○ Remove()

○ IndexOf()

○ Replace()

2. What is a constant variable?

○ A magic string or number.

○ A variable whose name must match its value.

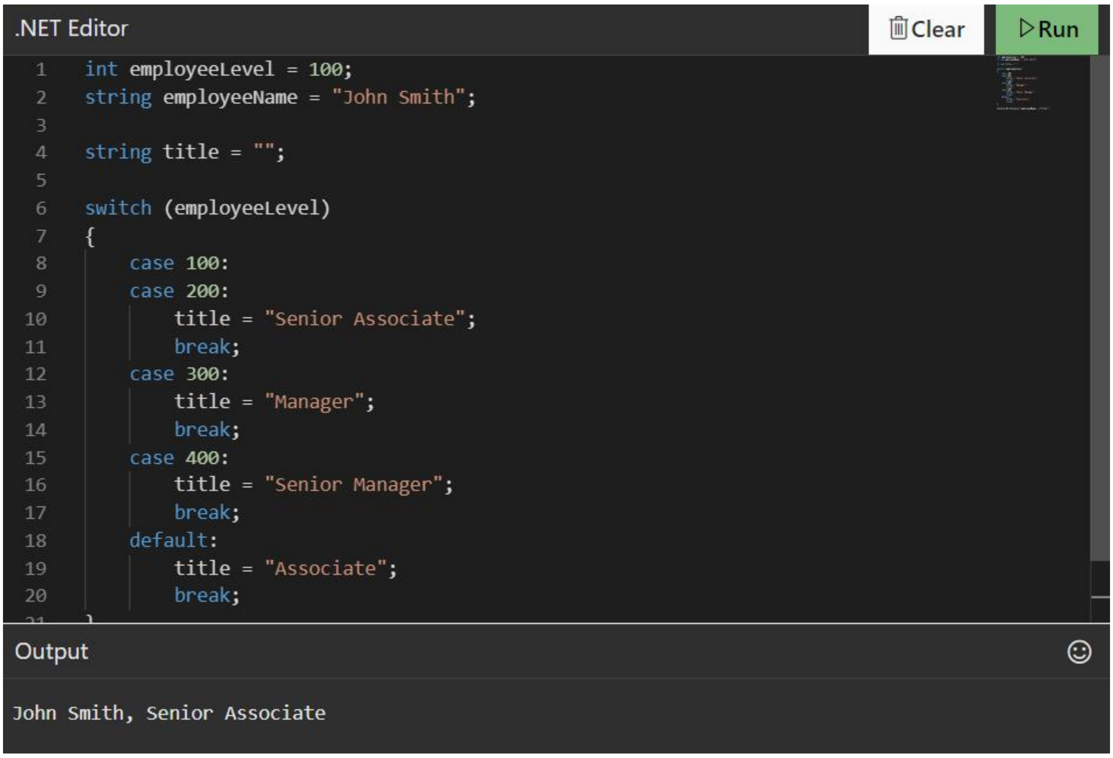
○ A variable that can only be used in the scope of the method.

● A value that once initialized can never be changed. ✓

Correct!

SESSION -5 :

Exercise - Use the switch case statement

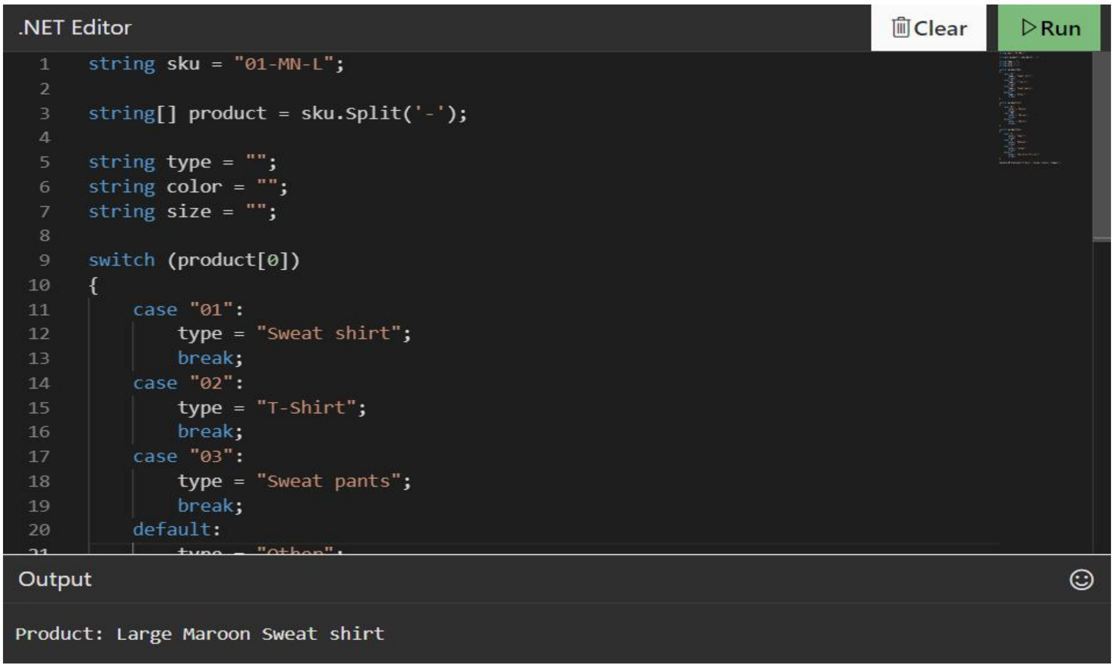


```csharp
int employeeLevel = 100;
string employeeName = "John Smith";

string title = "";

switch (employeeLevel)
{
    case 100:
    case 200:
        title = "Senior Associate";
        break;
    case 300:
        title = "Manager";
        break;
    case 400:
        title = "Senior Manager";
        break;
    default:
        title = "Associate";
        break;
}
```

```
John Smith, Senior Associate
```

Challenge

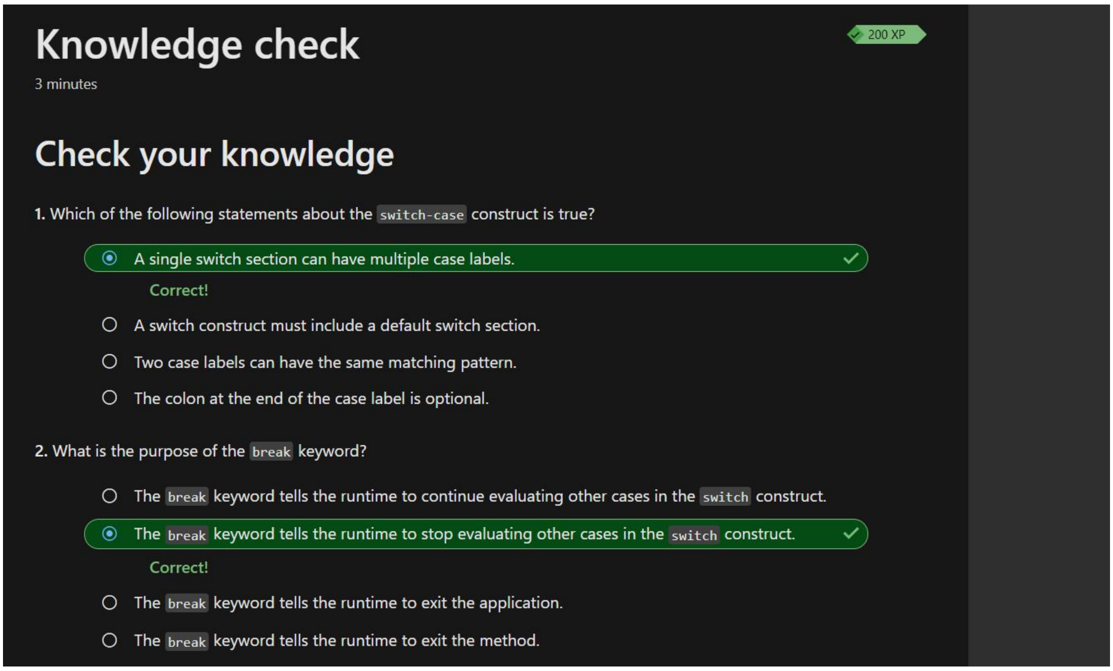
Look up product by SKU challenge



```csharp
string sku = "01-MN-L";

string[] product = sku.Split('-');

string type = "";
string color = "";
string size = "";

switch (product[0])
{
    case "01":
        type = "Sweat shirt";
        break;
    case "02":
        type = "T-Shirt";
        break;
    case "03":
        type = "Sweat pants";
        break;
    default:
        type = "Other";
```

```
Product: Large Maroon Sweat shirt
```

MCQ :



Knowledge check

3 minutes

Check your knowledge

1. Which of the following statements about the `switch-case` construct is true?

● A single switch section can have multiple case labels. ✓

Correct!

○ A switch construct must include a default switch section.

○ Two case labels can have the same matching pattern.

○ The colon at the end of the case label is optional.

2. What is the purpose of the `break` keyword?

○ The `break` keyword tells the runtime to continue evaluating other cases in the `switch` construct.

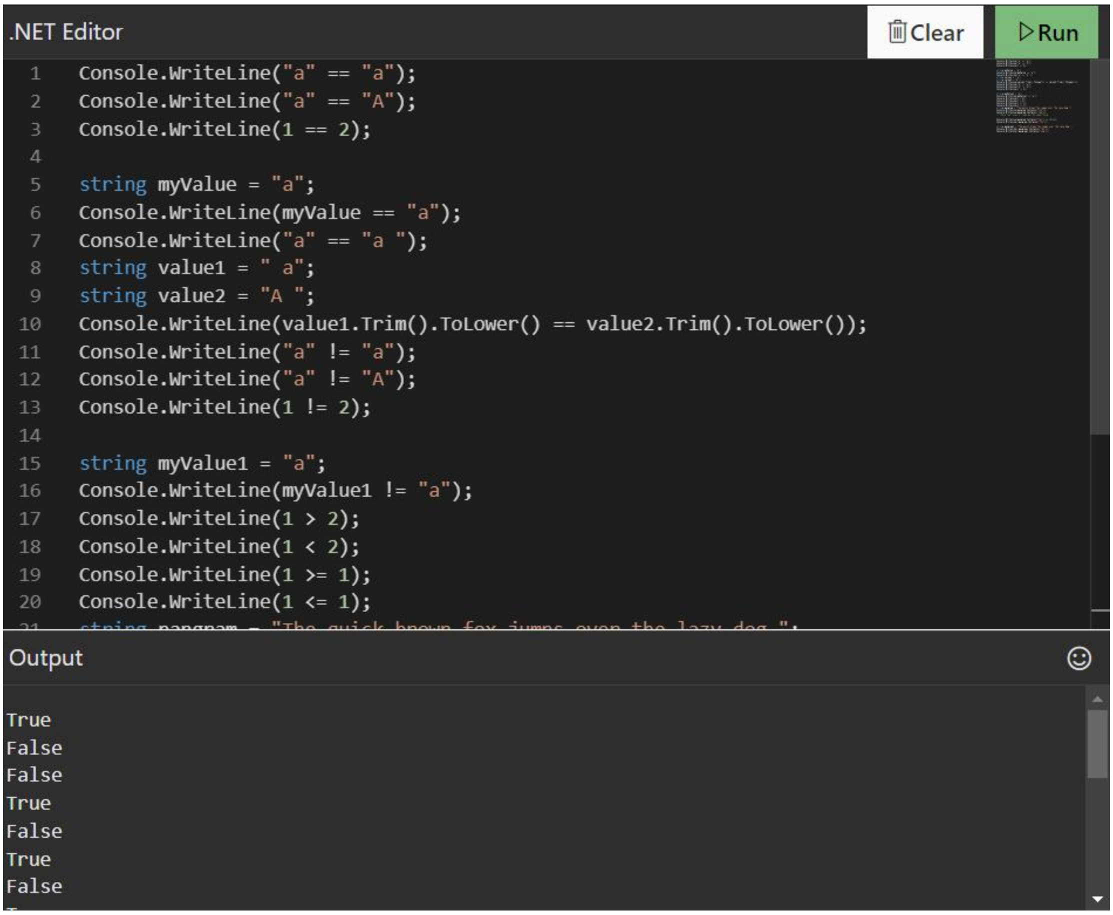
● The `break` keyword tells the runtime to stop evaluating other cases in the `switch` construct. ✓

Correct!

○ The `break` keyword tells the runtime to exit the application.

○ The `break` keyword tells the runtime to exit the method.

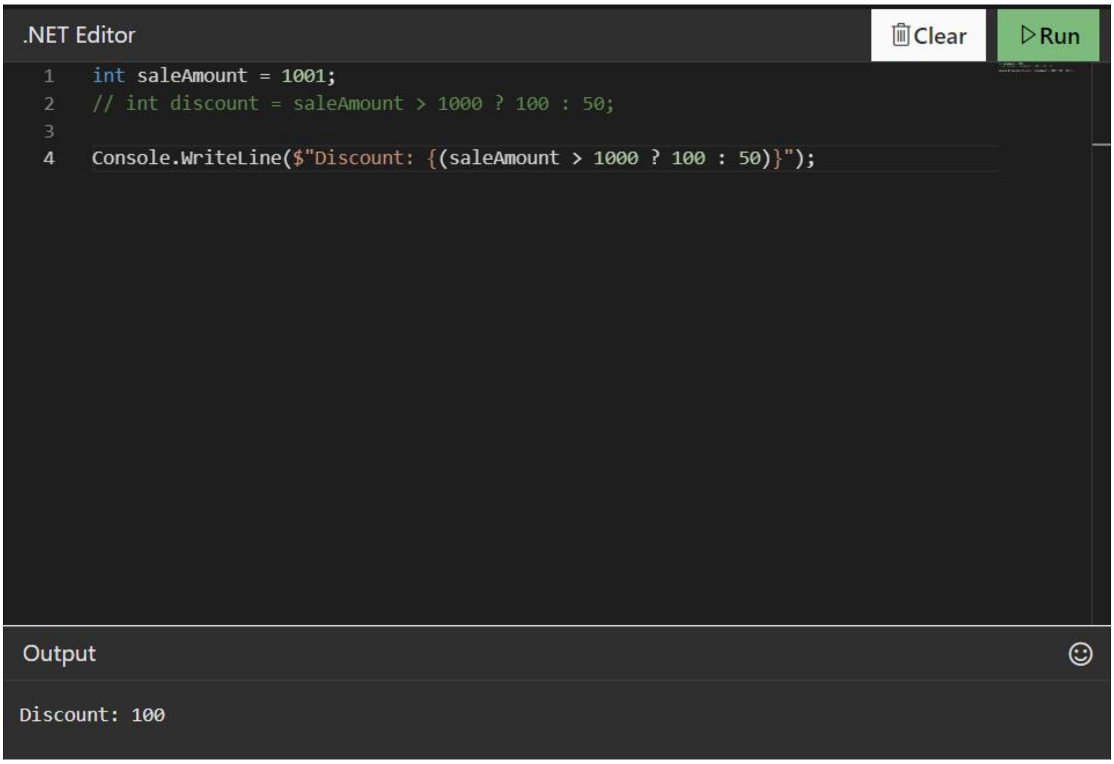
Exercise - Boolean Expressions



```
Console.WriteLine("a" == "a");
Console.WriteLine("a" == "A");
Console.WriteLine(1 == 2);

string myValue = "a";
Console.WriteLine(myValue == "a");
Console.WriteLine("a" == "a ");
string value1 = " a";
string value2 = "A ";
Console.WriteLine(value1.Trim().ToLower() == value2.Trim().ToLower());
Console.WriteLine("a" != "a");
Console.WriteLine("a" != "A");
Console.WriteLine(1 != 2);

string myValue1 = "a";
Console.WriteLine(myValue1 != "a");
Console.WriteLine(1 > 2);
Console.WriteLine(1 < 2);
Console.WriteLine(1 >= 1);
Console.WriteLine(1 <= 1);
string pangram = "The quick brown fox jumps over the lazy dog ";
```

```
True
False
False
True
False
True
False
```

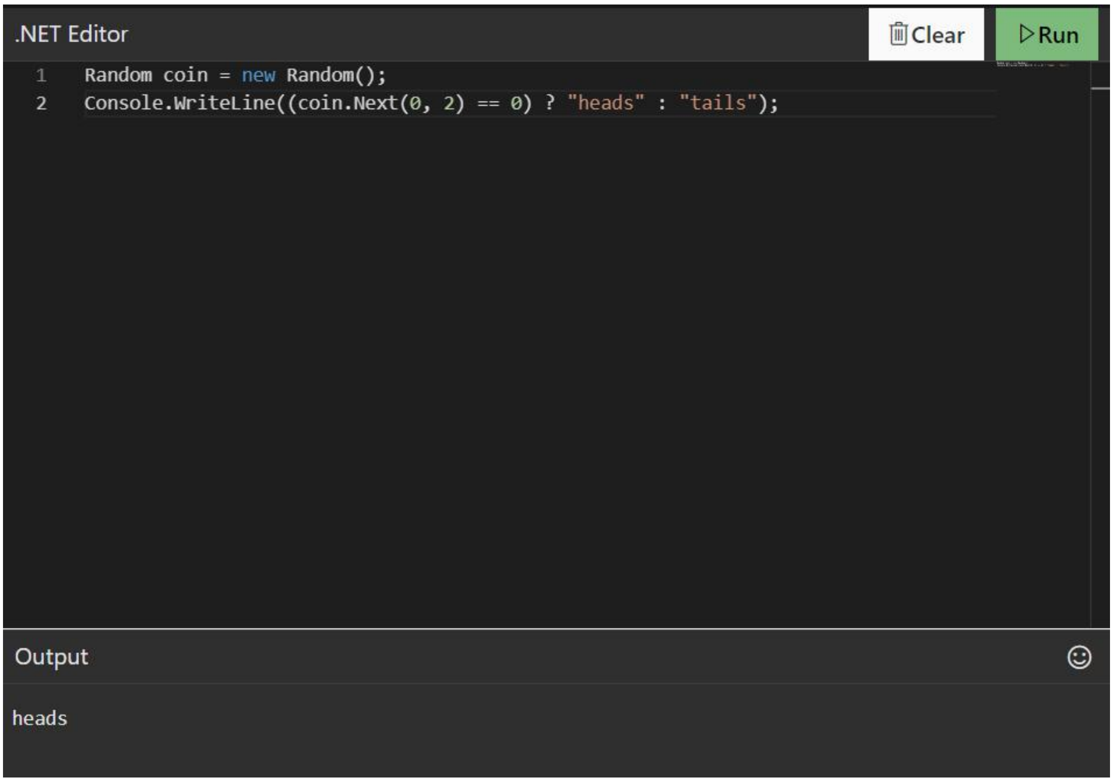
Exercise - conditional operator



```
int saleAmount = 1001;
// int discount = saleAmount > 1000 ? 100 : 50;

Console.WriteLine($"Discount: {(saleAmount > 1000 ? 100 : 50)}");
```

```
Discount: 100
```

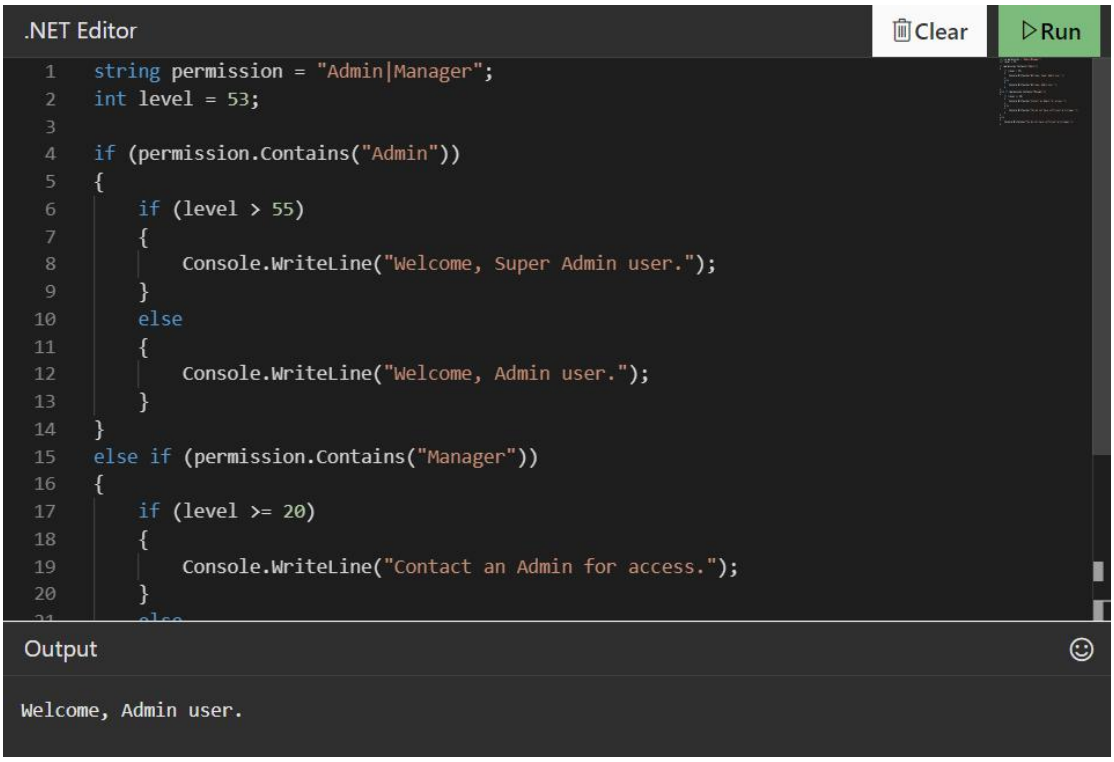
Heads or tails challenge



```
Random coin = new Random();
Console.WriteLine((coin.Next(0, 2) == 0) ? "heads" : "tails");
```

```
heads
```

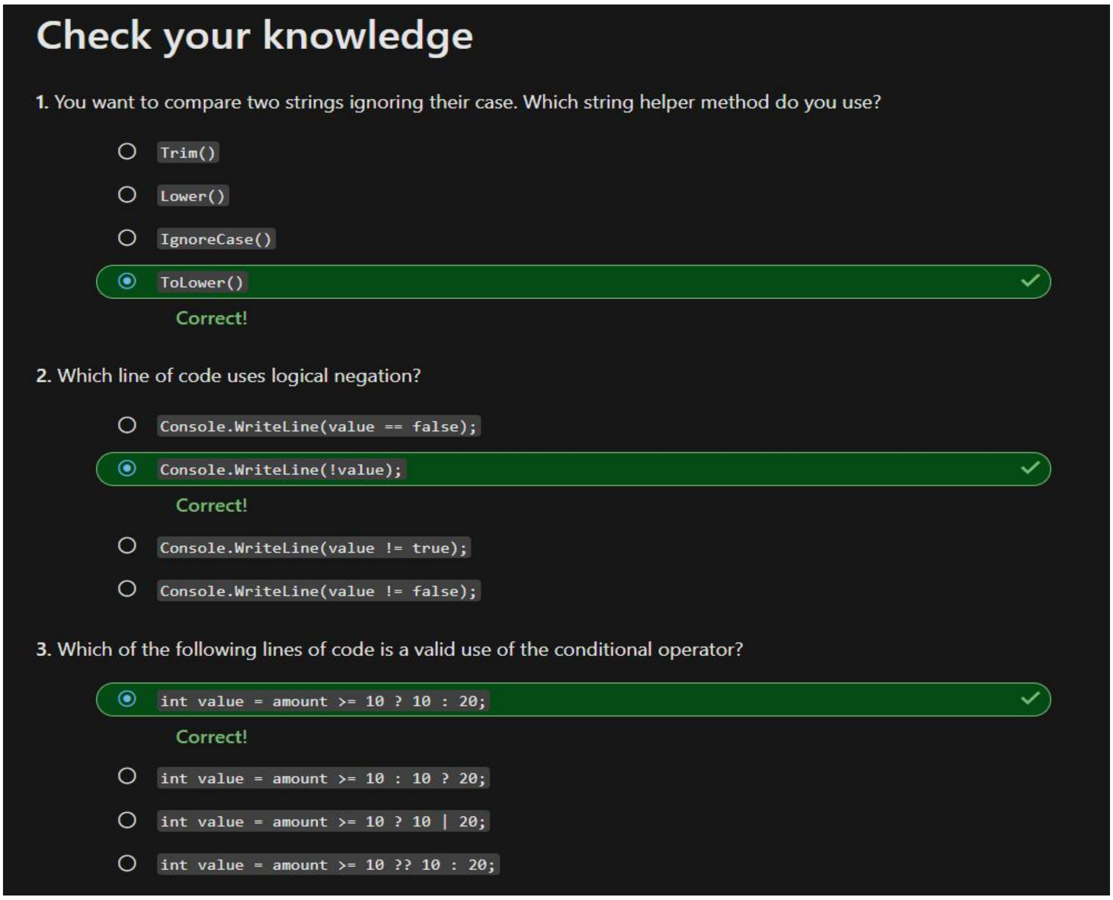
Complicated Permissions Challenge



```
string permission = "Admin|Manager";
int level = 53;

if (permission.Contains("Admin"))
{
    if (level > 55)
    {
        Console.WriteLine("Welcome, Super Admin user.");
    }
    else
    {
        Console.WriteLine("Welcome, Admin user.");
    }
}
else if (permission.Contains("Manager"))
{
    if (level >= 20)
    {
        Console.WriteLine("Contact an Admin for access.");
    }
    else
```

```
Welcome, Admin user.
```

MCQ :



Check your knowledge

1. You want to compare two strings ignoring their case. Which string helper method do you use?

○ `Trim()`

○ `Lower()`

○ `IgnoreCase()`

● `ToLower()` ✓

Correct!

2. Which line of code uses logical negation?

○ `Console.WriteLine(value == false);`

● `Console.WriteLine(!value);` ✓

Correct!

○ `Console.WriteLine(value != true);`

○ `Console.WriteLine(value != false);`

3. Which of the following lines of code is a valid use of the conditional operator?

● `int value = amount >= 10 ? 10 : 20;` ✓

Correct!

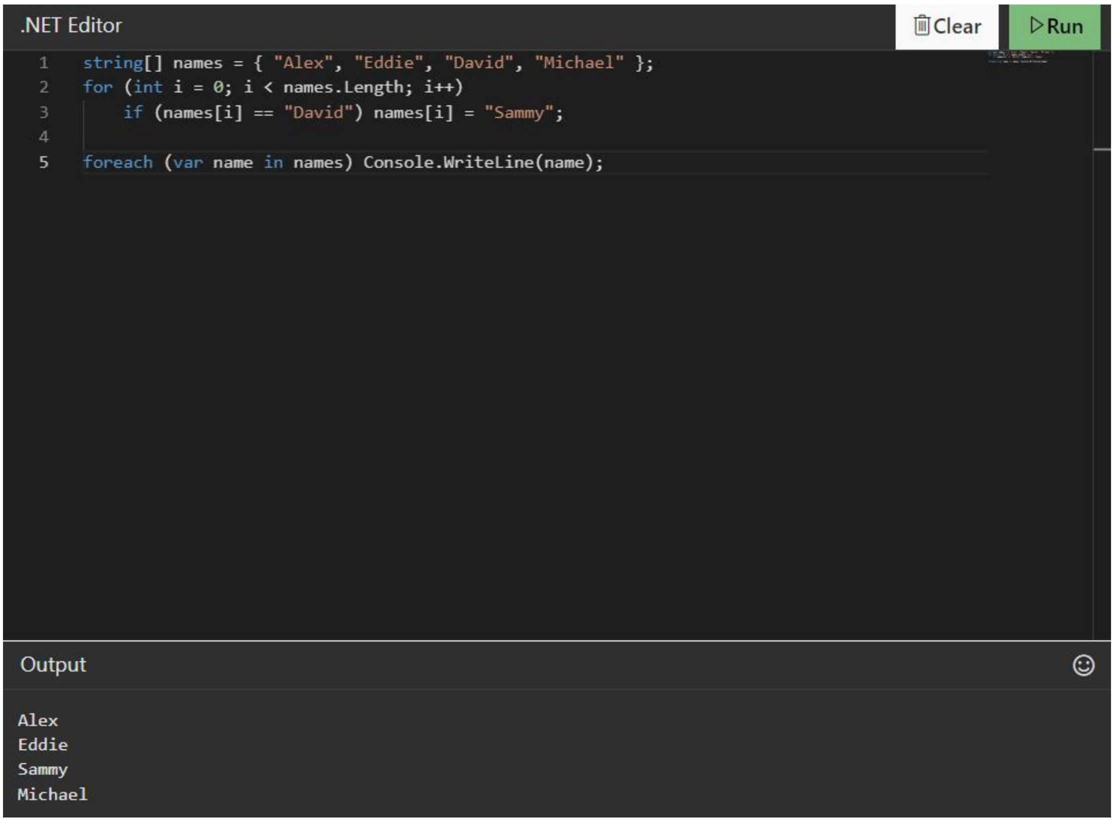
○ `int value = amount >= 10 : 10 ? 20;`

○ `int value = amount >= 10 ? 10 | 20;`

○ `int value = amount >= 10 ?? 10 : 20;`

SESSION -6 :

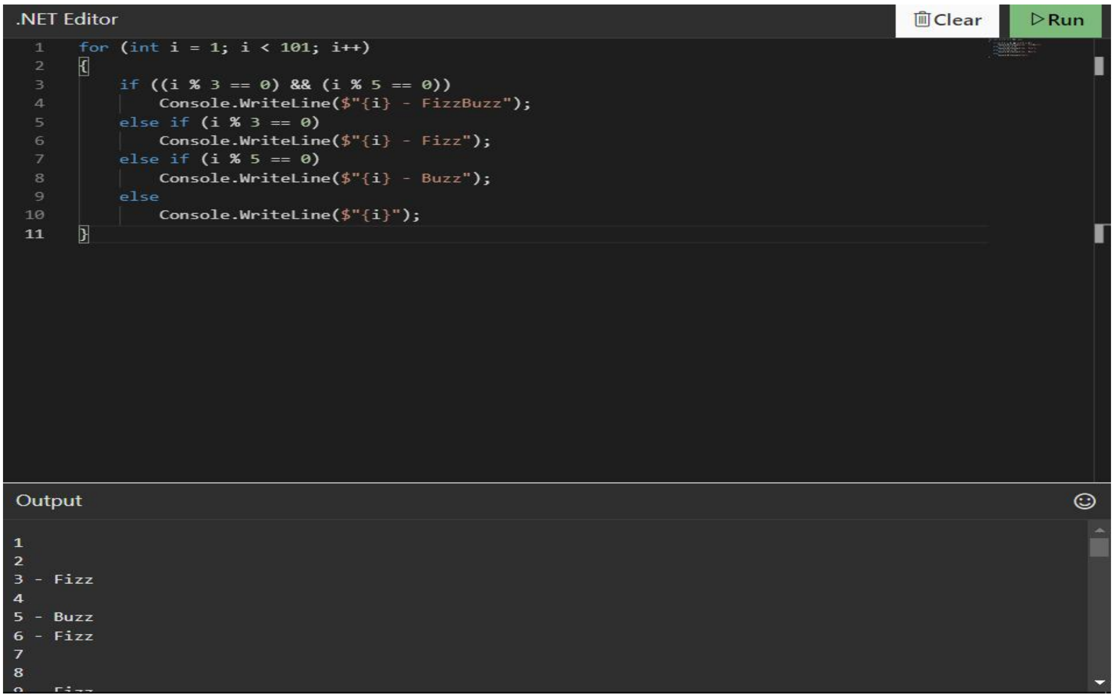
Exercise - for iteration statement



```
string[] names = { "Alex", "Eddie", "David", "Michael" };
for (int i = 0; i < names.Length; i++)
    if (names[i] == "David") names[i] = "Sammy";

foreach (var name in names) Console.WriteLine(name);
```

```
Alex
Eddie
Sammy
Michael
```

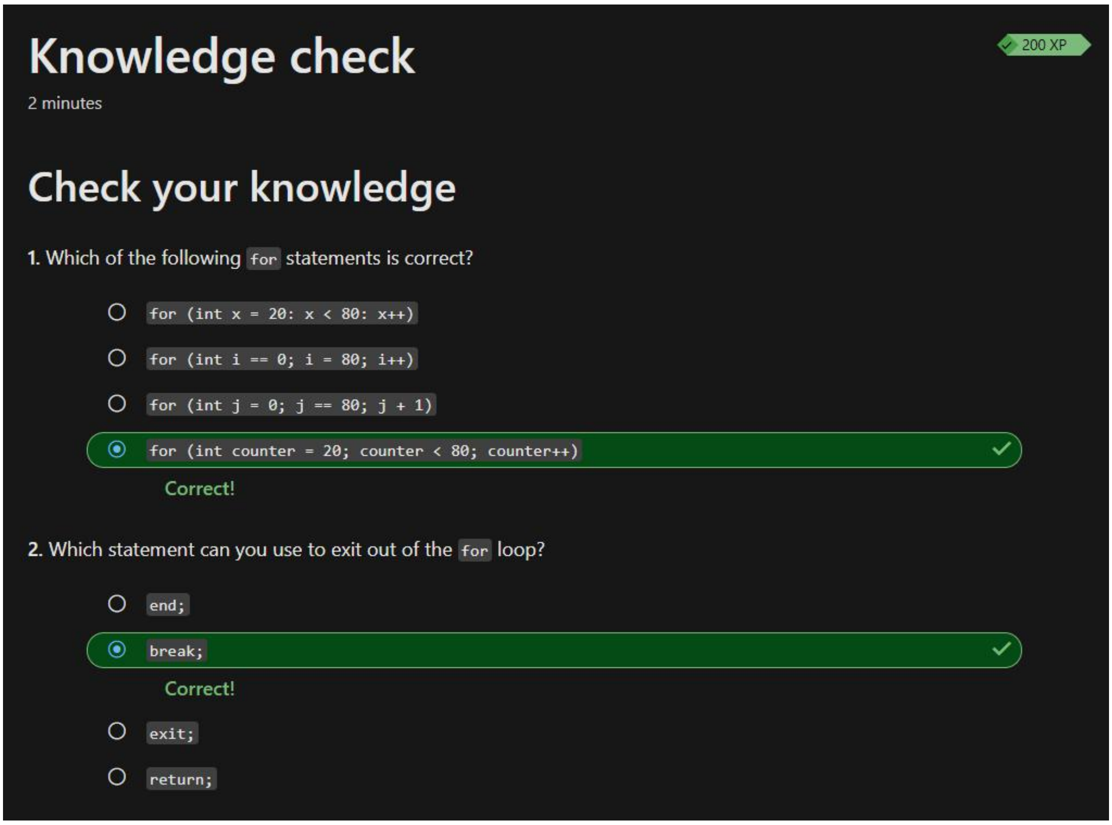
FizzBuzz challenge



```
for (int i = 1; i < 101; i++)
{
    if ((i % 3 == 0) && (i % 5 == 0))
        Console.WriteLine($"{i} - FizzBuzz");
    else if (i % 3 == 0)
        Console.WriteLine($"{i} - Fizz");
    else if (i % 5 == 0)
        Console.WriteLine($"{i} - Buzz");
    else
        Console.WriteLine($"{i}");
}
```

```
1
2
3 - Fizz
4
5 - Buzz
6 - Fizz
7
8
9 - Fizz
```

MCQ :



Knowledge check

2 minutes

Check your knowledge

1. Which of the following `for` statements is correct?

○ `for (int x = 20: x < 80: x++)`

○ `for (int i == 0; i = 80; i++)`

○ `for (int j = 0; j == 80; j + 1)`

● `for (int counter = 20; counter < 80; counter++)` ✓

Correct!

2. Which statement can you use to exit out of the `for` loop?

○ `end;`

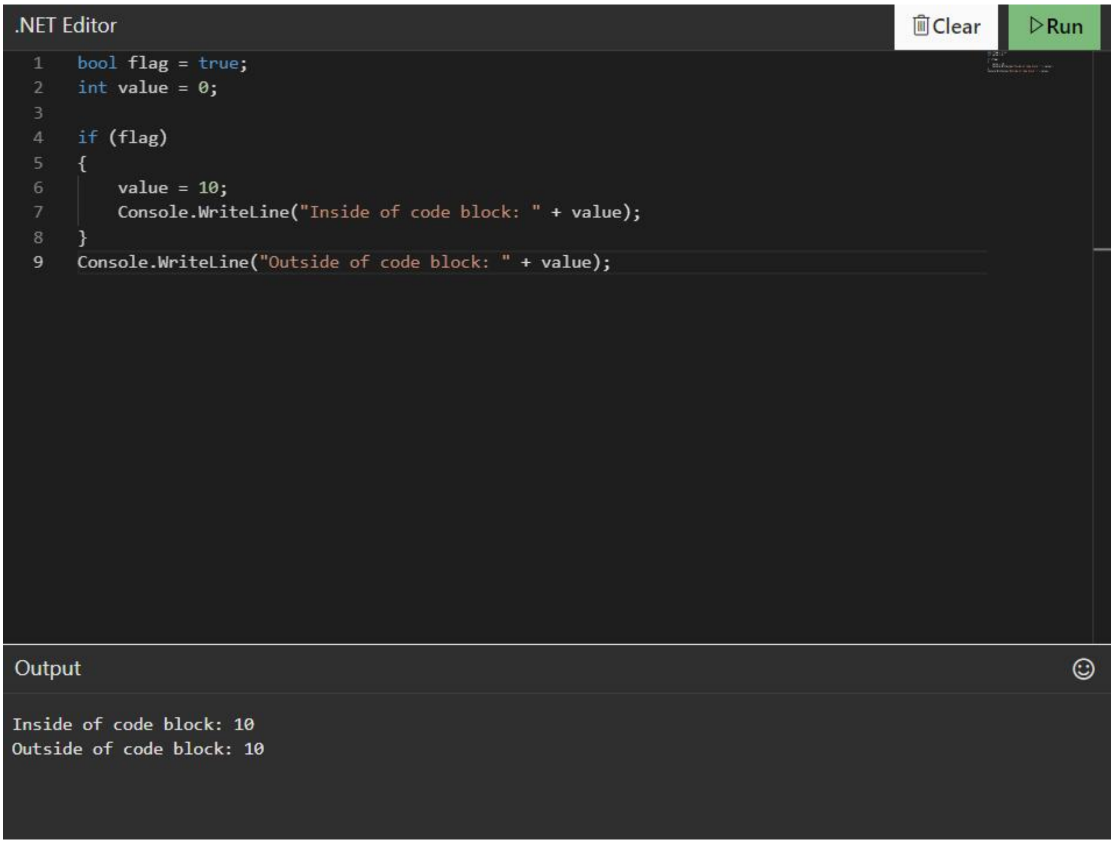
● `break;` ✓

Correct!

○ `exit;`

○ `return;`

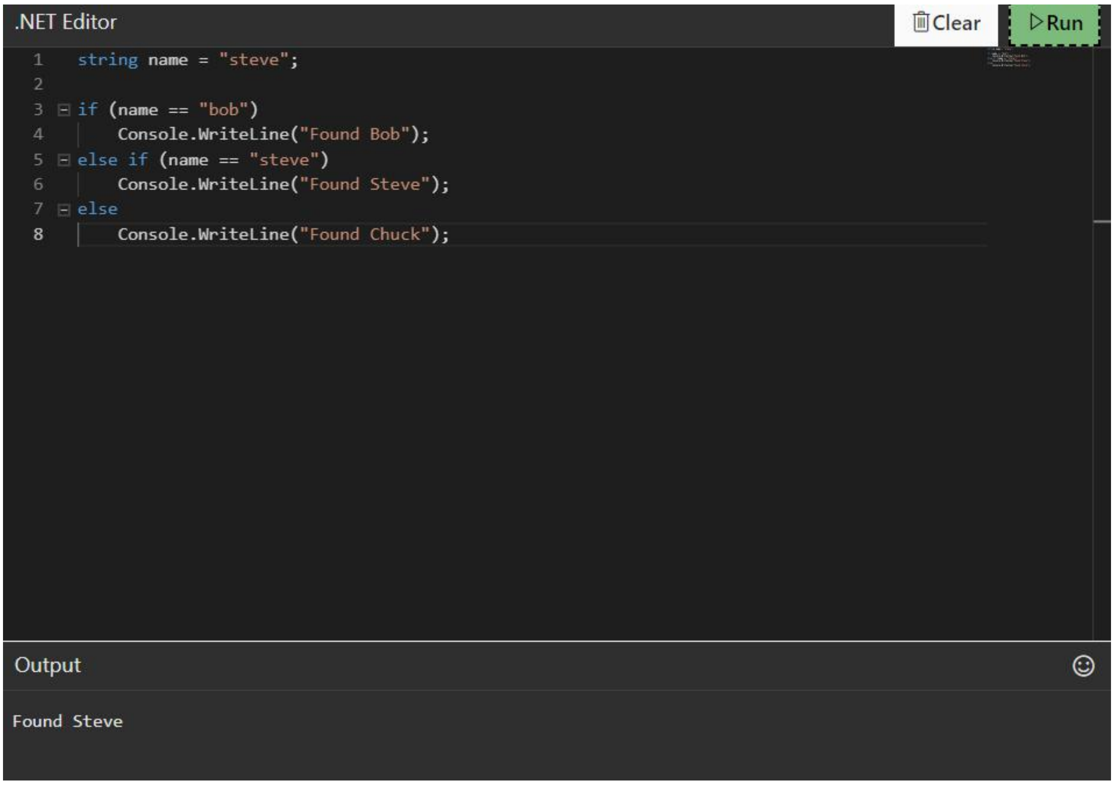
Exercise - Code blocks and variable scope



```
bool flag = true;
int value = 0;

if (flag)
{
    value = 10;
    Console.WriteLine("Inside of code block: " + value);
}
Console.WriteLine("Outside of code block: " + value);
```

```
Inside of code block: 10
Outside of code block: 10
```

Exercise - Remove code blocks in if statements



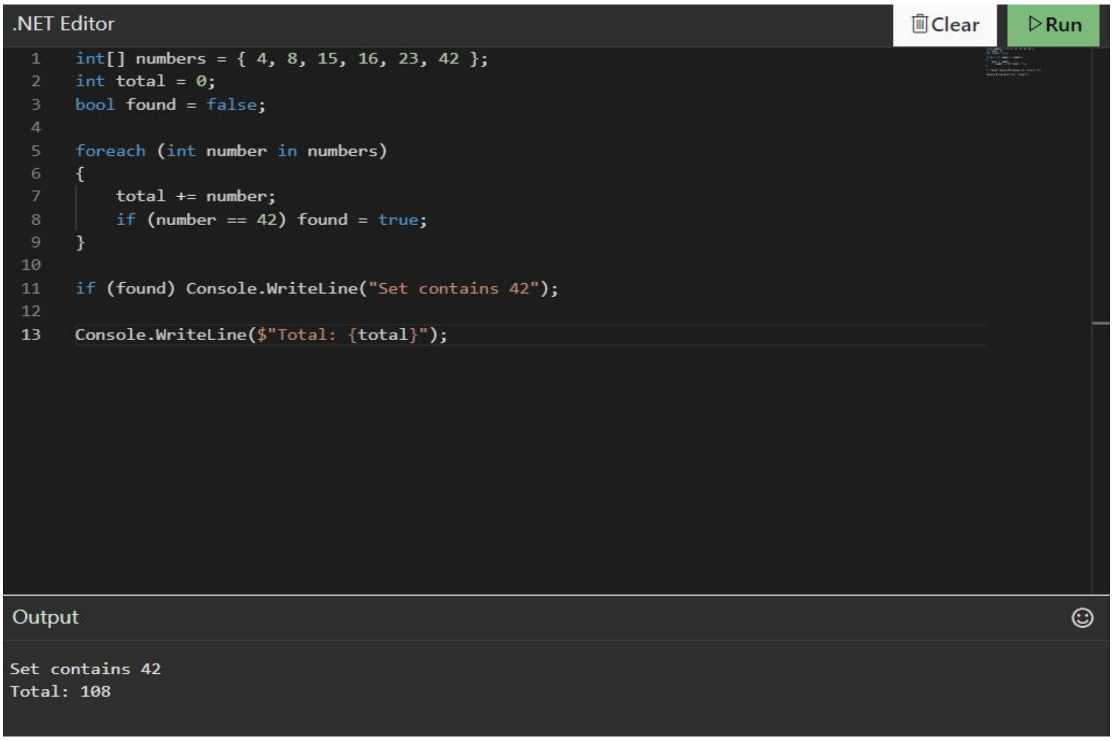
```
string name = "steve";

if (name == "bob")
    Console.WriteLine("Found Bob");
else if (name == "steve")
    Console.WriteLine("Found Steve");
else
    Console.WriteLine("Found Chuck");
```

```
Found Steve
```

Challenge

Fix this code

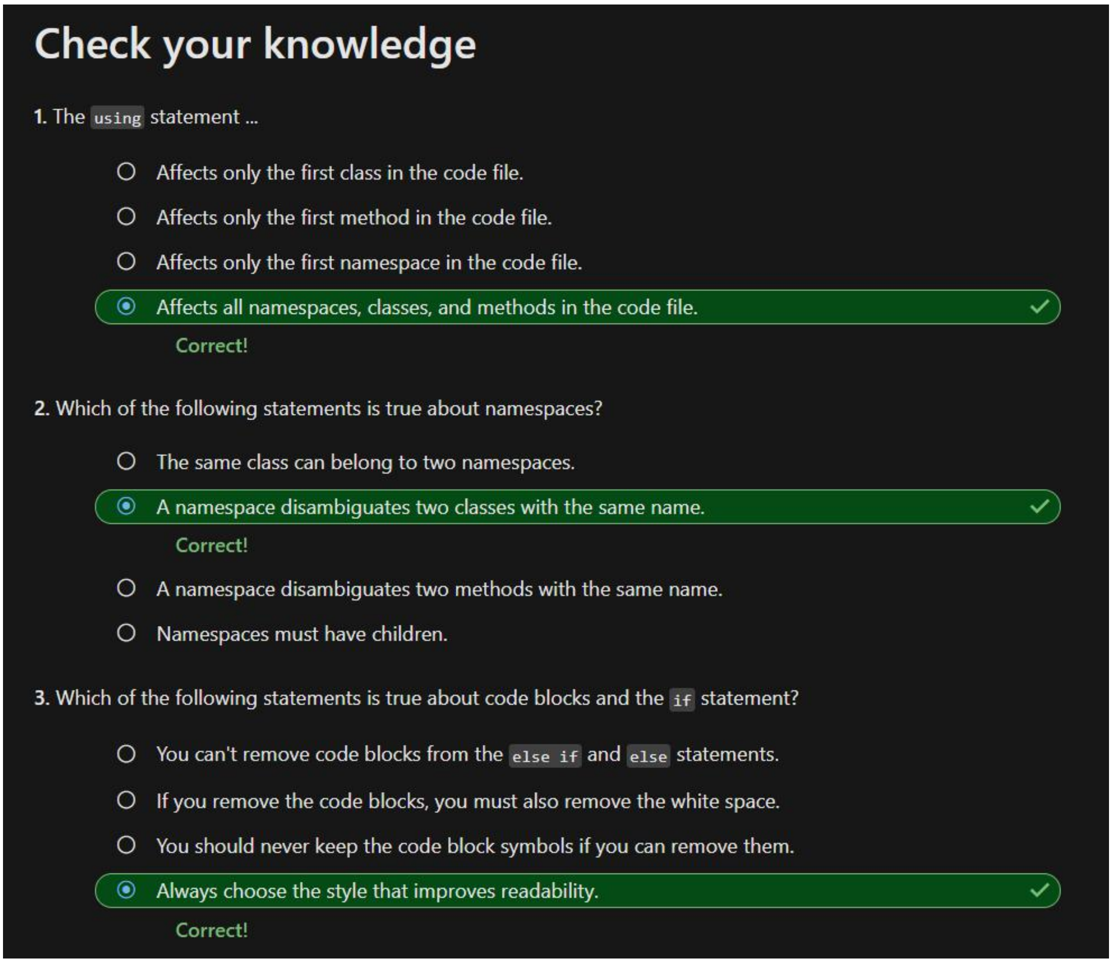
Use what you've learned in this module to fix this poorly written code. There are many improvements that you can make.



```
int[] numbers = { 4, 8, 15, 16, 23, 42 };
int total = 0;
bool found = false;

foreach (int number in numbers)
{
    total += number;
    if (number == 42) found = true;
}

if (found) Console.WriteLine("Set contains 42");

Console.WriteLine($"Total: {total}");
```

```
Set contains 42
Total: 108
```

MCQ :



Check your knowledge

1. The `using` statement ...

- ○ Affects only the first class in the code file.
- ○ Affects only the first method in the code file.
- ○ Affects only the first namespace in the code file.
- ● Affects all namespaces, classes, and methods in the code file. ✓

  Correct!

2. Which of the following statements is true about namespaces?

- ○ The same class can belong to two namespaces.
- ● A namespace disambiguates two classes with the same name. ✓

  Correct!

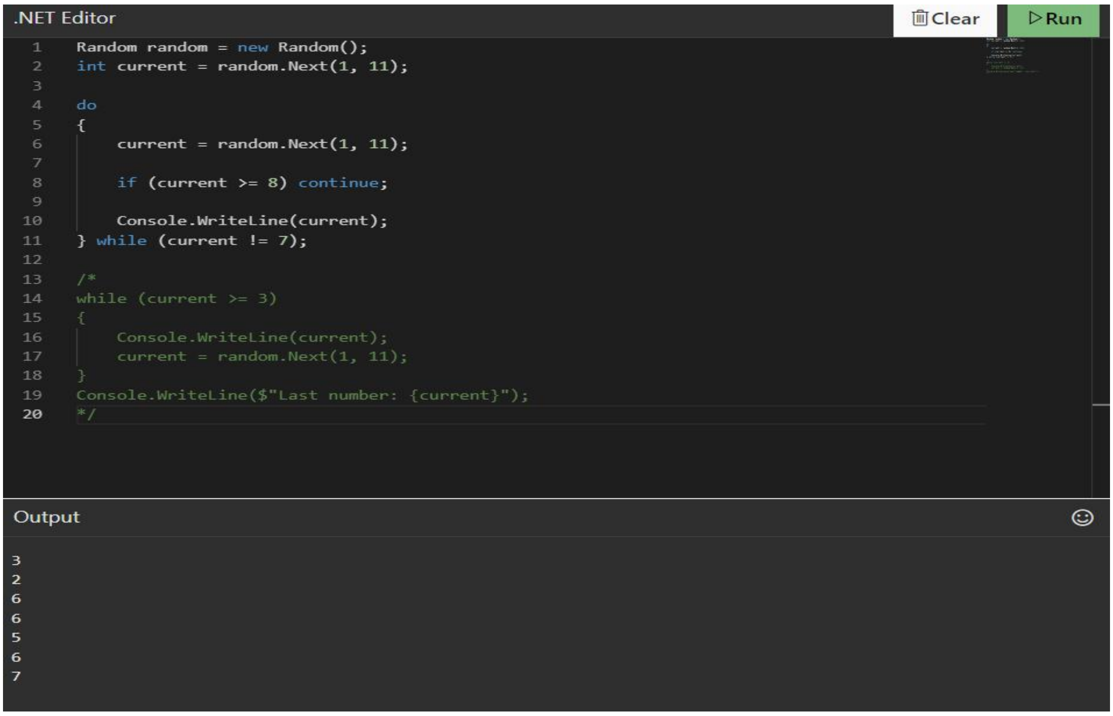
- ○ A namespace disambiguates two methods with the same name.
- ○ Namespaces must have children.

3. Which of the following statements is true about code blocks and the `if` statement?

- ○ You can't remove code blocks from the `else if` and `else` statements.
- ○ If you remove the code blocks, you must also remove the white space.
- ○ You should never keep the code block symbols if you can remove them.
- ● Always choose the style that improves readability. ✓

  Correct!

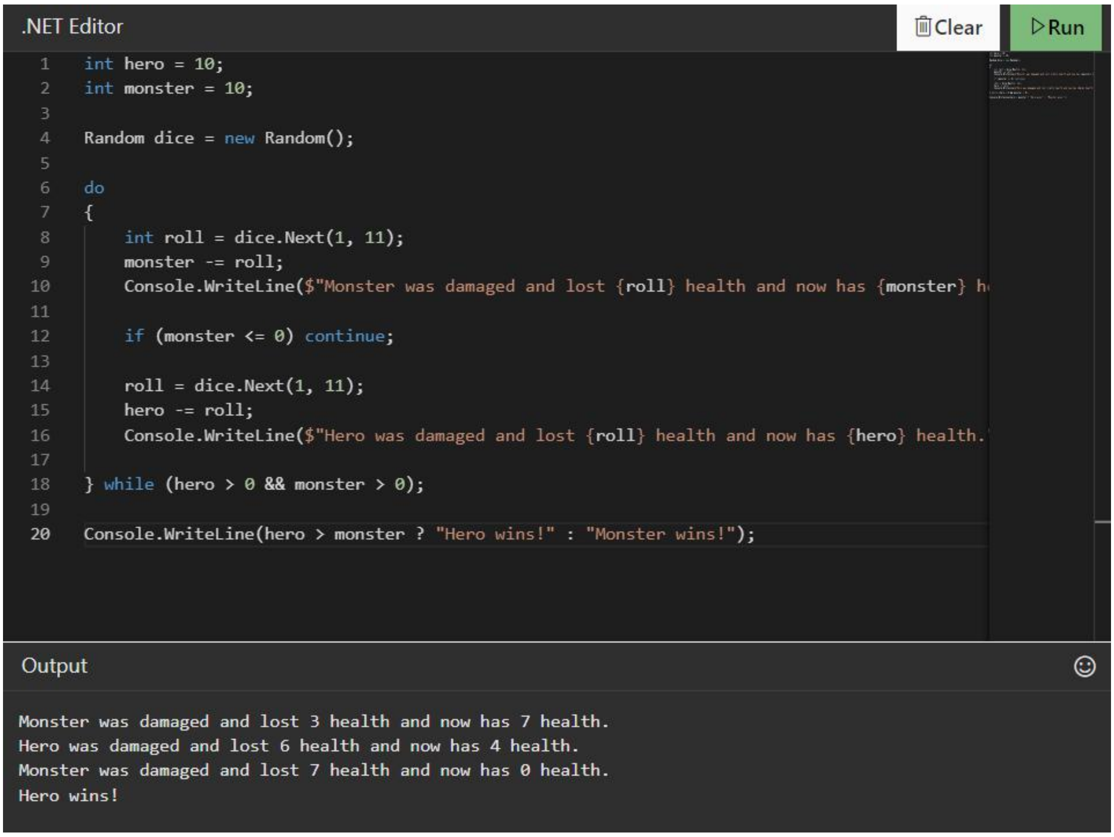
Exercise - do-while, while, and continue statements



```
Random random = new Random();
int current = random.Next(1, 11);

do
{
    current = random.Next(1, 11);

    if (current >= 8) continue;

    Console.WriteLine(current);
} while (current != 7);

/*
while (current >= 3)
{
    Console.WriteLine(current);
    current = random.Next(1, 11);
}
Console.WriteLine($"Last number: {current}");
*/
```

```
3
2
6
6
5
6
7
```

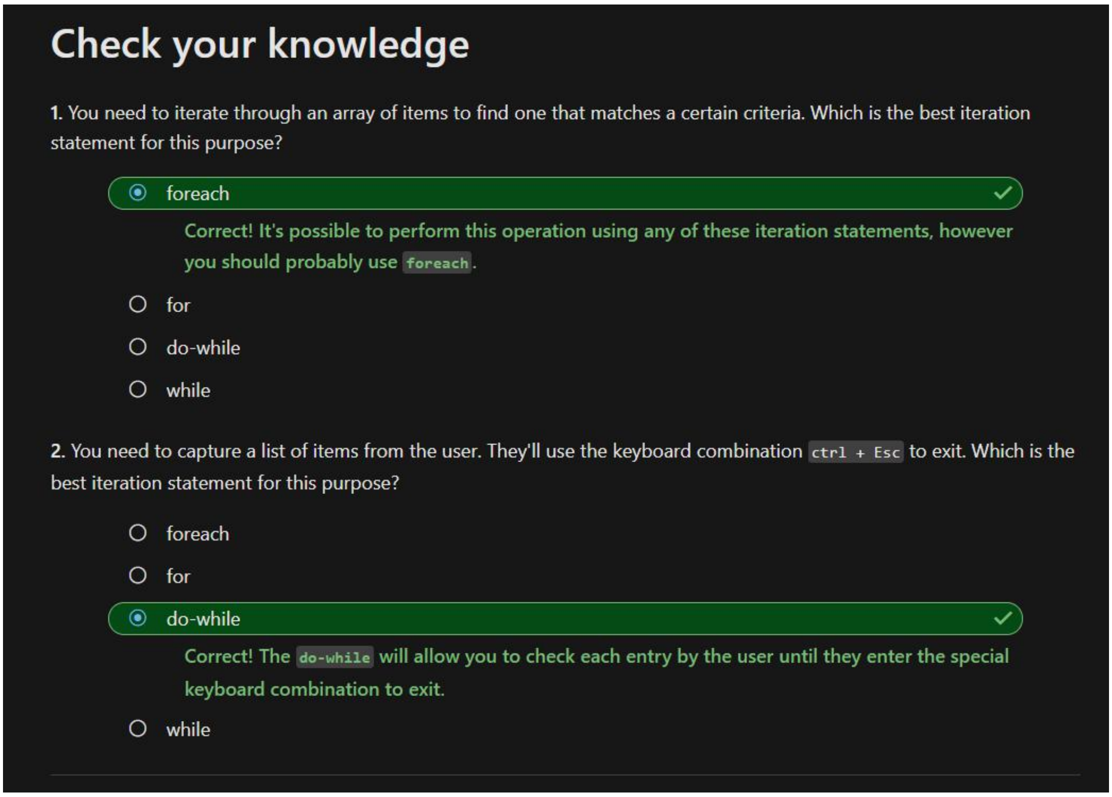
Role playing game battle challenge



```csharp
int hero = 10;
int monster = 10;

Random dice = new Random();

do
{
    int roll = dice.Next(1, 11);
    monster -= roll;
    Console.WriteLine($"Monster was damaged and lost {roll} health and now has {monster} h

    if (monster <= 0) continue;

    roll = dice.Next(1, 11);
    hero -= roll;
    Console.WriteLine($"Hero was damaged and lost {roll} health and now has {hero} health.

} while (hero > 0 && monster > 0);

Console.WriteLine(hero > monster ? "Hero wins!" : "Monster wins!");
```

```
Monster was damaged and lost 3 health and now has 7 health.
Hero was damaged and lost 6 health and now has 4 health.
Monster was damaged and lost 7 health and now has 0 health.
Hero wins!
```

MCQ :



Check your knowledge

1. You need to iterate through an array of items to find one that matches a certain criteria. Which is the best iteration statement for this purpose?

- ● foreach ✓

  Correct! It's possible to perform this operation using any of these iteration statements, however you should probably use `foreach`.

- ○ for
- ○ do-while
- ○ while

2. You need to capture a list of items from the user. They'll use the keyboard combination `ctrl + Esc` to exit. Which is the best iteration statement for this purpose?

- ○ foreach
- ○ for
- ● do-while ✓

  Correct! The `do-while` will allow you to check each entry by the user until they enter the special keyboard combination to exit.

- ○ while