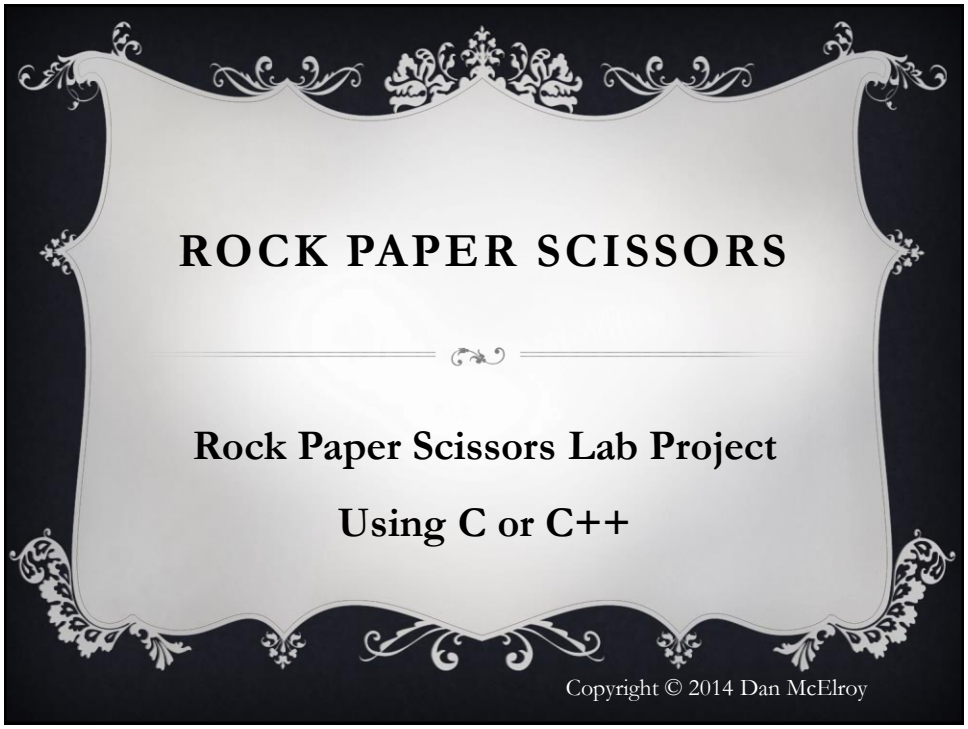

Topics Covered

- Project Definition
- Truth Tables
- Keyboard Character Input (C and C++)
- Converting Characters to Uppercase for easier processing
- Character Literals (constants) vs. Variables
- Use a Loop to Repeat the Game

Project Definition

Write a program to score the rock-paper-scissor game. Each of two users types in either R,P, or S.

The program then announces the winner as well as the basis for determining the winner:
Paper covers rock, Rock breaks scissors, Scissors cut paper, or a Tie game.

Be sure to allow the user to use lowercase as well as uppercase letters. Your program should include a loop that lets the user play again until the user says she or he is done.

Combinations to Determine the Winner

There are nine conditions to test for:

Player 1	Player 2	Who wins?
R	R	Tie – Nobody wins
R	P	Player 2 – Paper covers rock
R	S	Player 1 – Rock breaks scissors
P	R	Player 1 – Paper covers rock
P	P	Tie – Nobody wins
P	S	Player 2 – Scissors cut paper
S	R	Player 2 – Rock breaks scissors
S	P	Player 1 – Scissors cut paper
S	S	Tie – Nobody wins

Truth-Table

A Truth-Table is used to simplify the design and to make sure that all conditions are tested.

	Player2		
	R	P	S
Player1			
R	Tie	Player2 wins Paper covers Rock	Player1 wins Rock breaks Scissors
P	Player1 wins Paper covers Rock	Tie	Player2 wins Scissors cut Paper
S	Player2 wins Rock breaks Scissors	Player1 wins Scissors cut Paper	Tie

Truth-Table

If the selection for Player1 is equal to the selection for Player2, it is a tie. Six conditions remain.

	Player2		
	R	P	S
Player1			
R	Tie	Player2 wins Paper covers Rock	Player1 wins Rock breaks Scissors
P	Player1 wins Paper covers Rock	Tie	Player2 wins Scissors cut Paper
S	Player2 wins Rock breaks Scissors	Player1 wins Scissors cut Paper	Tie

Keyboard Character Input

The method for inputting a single character from the keyboard is different between C and C++.

The C-language uses scanf and printf to input and output data to the terminal. A newer, more secure version of scanf is called scanf_s.

The C++ language uses cin and cout to input and output data to the terminal.

C-language Character Input

Use the following code to input a character:

```
/* at the top of the program */
#include <stdio.h>
#include <ctype.h>

    /* in the middle of the program */
    char Player1;
    flushall();    /* clear out the keyboard input buffer */
    printf ("Player 1, enter R P or S: ");
    scanf_s ("%c", &Player1);  /* %c for character */
    Player1 = toupper(Player1);
```

When the code executes, the character from the keyboard will be placed in the variable **Player1**.

Do the same for Player2.

C++ language Character Input

Use the following code to input a character:

```
// at the top of the program
#include <iostream>
#include <cctype>
using namespace std;

    // in the middle of the program
    char Player1;
    cout << "Player 1, enter R P or S: ";
    cin  >> Player1;
    Player1 = toupper(Player1);
```

When the code executes, the character from the keyboard will be placed in the variable **Player1**.

Do the same for Player2.

Convert to Uppercase

The toupper( ) function was used to convert the user's input to uppercase.

```
Player1 = toupper(Player1);
```

Now it is only necessary to compare the user's input to an uppercase character.

```
if (Player1 == 'R' && Player2 == 'P')
    . . .
```

Checking for a Tie

If the input for Player1 is equal to the input for Player2, then it is a tie game. Don't forget the double equal signs to test for equality.

```
if (Player1 == Player2)
  // display a message that shows a tie
```

Literals (Constants) vs. Variables

In this example, Player1 and Player2 are the names of the variables that will hold the input from the user. These variables can be compared against a character literal (constant) to determine which character was input by the user.

```
char Player1;
if (Player1 == 'R' && Player2 == 'P')
  // display a message that shows who won
```

Literals (Constants) vs. Variables

In Example 1, the contents of the variable Player1 is compared against the character literal 'R'. This is correct. Note that character literals are defined by single quotes in C and C++. String literals are defined by double quotes.

Example 1:

```
if (Player1 == 'R' && Player2 == 'P')
  // display a message that shows who won
```

Literals (Constants) vs. Variables

In Example 2, the contents of the variable Player1 is compared against the contents of the variable named R. This variable must be defined and the contents of the variable must be set to something before the comparison is made.

Example 2:

```
if (Player1 == R && Player2 == P)
```

Input Validation

Your program still needs to verify that only the characters R P or S were entered by the user. Display a message indicating illegal input if anything except those three characters were entered.

C++ Use a Loop to Repeat the Game

The easiest way to set up the loop is to create it before you put in the logic for the game itself. The next slide shows sample code for the C-language.

```
char playAgain = 'y';  // loop control
do
{
   // the logic for the game goes here

   cout << "Do you want to play again? ";
   cin  >> playAgain;
} while (playAgain=='Y' || playAgain=='y');
```

C Use a Loop to Repeat the Game

The easiest way to set up the loop is to create it before you put in the logic for the game itself. Your compiler may need scanf_s instead of scanf.

```
char playAgain = 'y';  // loop control
do
{
   // the logic for the game goes here

   printf ("Do you want to play again? ";
   scanf ("% c", &playAgain); // note the space
} while (playAgain=='Y' || playAgain=='y');
```