

Flow of Control

- ❖ **Flow of control**
 - ◆ **Definition: The sequence in which the computer executes statements of the program.**
- ❖ **Sequential Control Structure**
- ❖ **Selection (Branching) Control Structure**
 - ◆ **Relational and Logical Operators**
- ❖ **Repetition (Loop) Control Structure**
 - ◆ **while loops**
 - ◆ **for loops**

Copyright © 2005 R.M. Laurie 1

The repetition control structure is used to repeat a sequence of statements when an assertion is true. It is the third of the three primary control structures that exist in all programming languages: Sequential, Selection, and Repetition.

Repetition structures utilize assertions very similar to the way that the "if" selection structures use assertions. If the assertion is true then a block of statements defined by the { } brackets are executed. If the assertion is false you exit the loop and skip the block of statements. The fundamental difference with selection structures is that after the block of statements is executed with a true assertion, then flow of control returns to the assertion and it is tested again. This results in a looping or repetition control structure that is best utilized for repeated tasks.

Repetition (Loop) Structure

- ❖ **Control structure used to repeat a sequence of instructions in a loop.**
- ❖ **The simplest loop structure is the while()**

```

graph TD
    Start([Start]) --> Init[Count = 1  
Sum = 0]
    Init --> Decision{Count <= 5}
    Decision -- True --> Prompt[Prompt: Score]
    Prompt --> SumAdd[Sum = Sum + Score]
    SumAdd --> DisplayScore[Display: Score]
    DisplayScore --> Increment[Increment Count]
    Increment --> Decision
    Decision -- False --> Avg[Avg = Sum/5]
    Avg --> DisplayAvg[Display: Avg]
    DisplayAvg --> End([End])
  
```

```

var Score, Count = 1, Sum = 0;
while(Count <= 5)
{
  Score = parseFloat(window.prompt(
    "Enter Score "+Count,"0"));
  Sum = Sum + Score;
  document.writeln("<p>Score "+Count
    +" = " +Score+"</p>");
  Count = Count + 1;
}
document.writeln("<h3>Average = "
  +(Sum/5)+"</h3>");
  
```

Copyright © 2005 R.M. Laurie 2

Examine the flow chart algorithm shown. This flow chart representation describes the repetition control structure that is used to prompt and input five exam scores and determine their average score. This algorithm describes two required variables Count and Sum.

Count contains the number of scores that the user has entered. This counter is initialized to one in the declaration statement. Note that the diamond shaped flow chart symbol contains the assertion that will be either true or false when determining if Count is less than or equal to 5. When this assertion is true the contents of the loop are executed. The last statement of the loop increments the Count and then program control returns back to the assertion evaluation. This looping results in the ability to repeat a sequence of statements while the assertions is true.

The Sum variable contains the running total of all entered scores. This sum variable is initialized to zero in the declaration statement. It is accessed in the loop to keep a running total of the all scores as they are entered.

The average of a group of scores is calculated to be the Sum of all scores divided by the total number of scores. For this reason there is no need to store all scores only their sum.

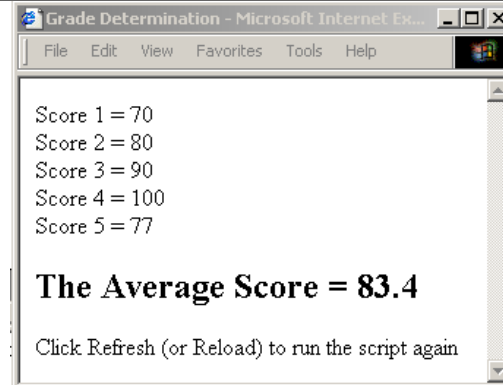
The results are displayed as follows: The scores are displayed as they are entered in the loop and the average is calculated when the loop assertion goes false and the loop is exited. Generally, one can think about a problem and program it by considering the sequence of events humans would do to solve the problem. Usually the most efficient sequence of events for humans to solve a problem is also typically the best way for the computer algorithm to be designed to solve a problem.

while statement loop control

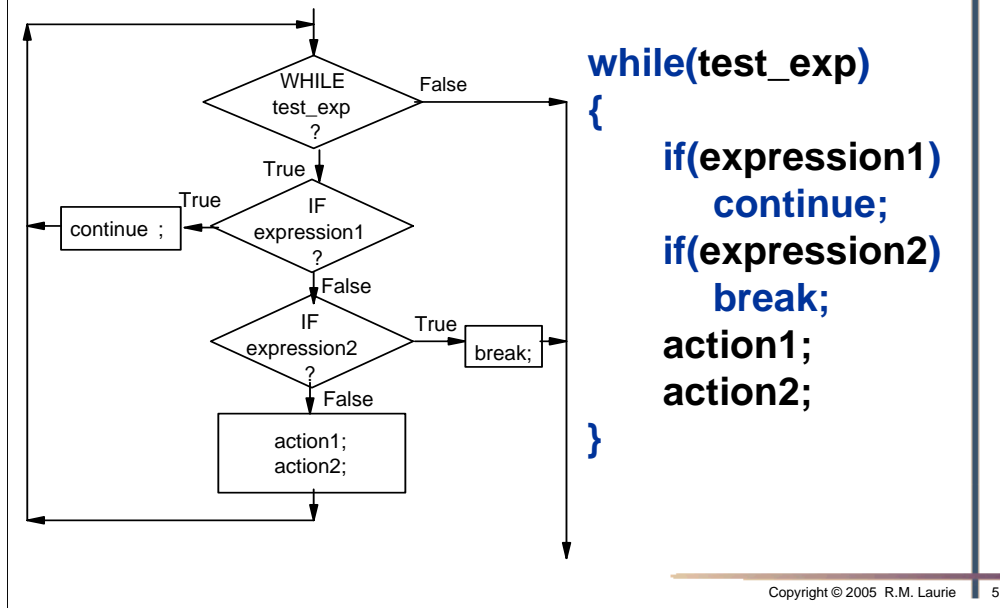
- ❖ Contents of loop executed repeatedly while(*assertion*) is **true**
- ❖ Loop terminated when while(*assertion*) is **false**.
- ❖ **Counter-Controlled Repetition Structure**
 - ◆ Initialize a counter to count loops
 - ◆ Increment or decrement counter
 - ◆ while(*assertion*) checks for total loops reached
- ❖ **Sentinel-Controlled Repetition Structure**
 - ◆ while(*assertion*) checks for a **sentinel** termination value

Counter-Controlled Repetition Structure

```
<head>
  <title>Grade Determination</title>
  <script type="text/javascript">
    var Score=0, ScoreTotal=0,
        Count=0;
    while(Count < 5)
    {
      Score=parseInt(window.prompt("Enter Score", ""));
      ScoreTotal = ScoreTotal + Score;
      Count = Count + 1;
      document.writeln("Score " + Count + " = " + Score + "<br/>");
    }
    document.writeln("<h2>The Average Score = " + ScoreTotal/5 + "</h2>");
  </script>
</head>
<body>
  <p>Click Refresh (or Reload) to run the script again</p>
</body>
```

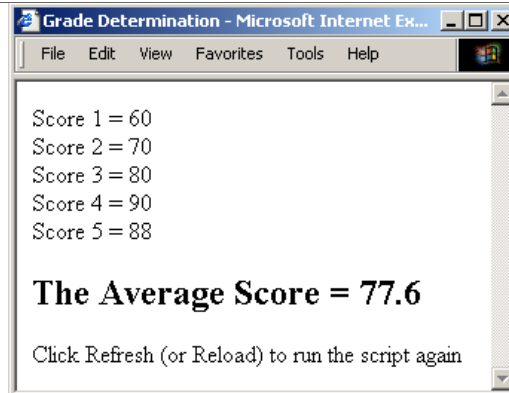


break; continue; commands



Sentinel-Controlled Repetition Structure

```
<head>
  <title>Grade Determination</title>
  <script type="text/javascript">
    var Score=0, ScoreTotal=0;
    var Count=0;
    while(true)
    {
      Score = parseFloat(window.prompt( "Enter Score (-1 to end)", "" ));
      if(Score < 0)
        break;
      ScoreTotal = ScoreTotal + Score;
      Count = Count + 1;
      document.writeln("Score " + Count + " = " + Score + "<br />");
    }
    document.writeln("<h2>The Average Score = "+ScoreTotal/Count+"</h2>");
  </script>
</head>
<body>  <p>Click Refresh (or Reload) to run the script again</p> </body>
```



Filtered Input Application

```
<head>
<title>Filtered Data Entry</title>
<script type="text/javascript">
  var Entry;
  while(true)
  {
    Entry = window.prompt( "Do you like Programming? (y or n)", "" );
    if(Entry == "y") {
      document.writeln("<h2>I'm glad you like programming!</h2>");
      break;
    }
    else if(Entry == "n") {
      document.writeln("<h2>You will like it if you study.</h2>");
      break;
    }
    else
      window.alert("You must enter either y or n !");
  }
</script>
</head>
<body> <p>Click Refresh (or Reload) to run the script again</p> </body>
```

Repetition Structure Exercise

- ❖ Create a program that will output the first 6 multiples of the number 7 with the format
 - ◆ $1 \times 7 = 7$
 - ◆ $2 \times 7 = 14 \dots$
- ❖ Create a program that will prompt for a base number and a maximum exponent. Output powers of that base from zero to the maximum exponent
 - ◆ $4^0 = 1$
 - ◆ $4^1 = 4$
 - ◆ $4^2 = 16 \dots$