# Lesson 4: Formatting Input Data for Arithmetic

**OBJECTIVES: In this lesson you will learn about**

- Data types
- String data versus numeric data
- How input data (from the prompt method) is stored as a string
- Why you need to format input data for arithmetic
- How to use built in JavaScript functions to format input data for arithmetic (parseInt, parseFloat, and eval)

## Preparing to Program

In Lesson 3 we learned how to use arithmetic to solve problems in JavaScript. In this lesson, you will take that one step further by using the prompt method to collect user data for your calculations. This is more complicated, because JavaScript (and most other languages) store input data in the form of a string, i.e. character data. In the Introduction, you learned that JavaScript has three basic data types, or formats for information: Numeric, String, and Boolean. Information in the form of a string cannot be used in arithmetic: it must be converted (formatted) into the numeric format that the computer can use to carry out arithmetic.

This might seem strange to you. Think of it this way: you have been using HTML to format your output. Sometimes you use bold; sometimes you use different colors. In a similar way, you need to format the data ***coming into*** your program. If you want to use input data to carry out arithmetic, you need to convert it to a numeric format.

The following example illustrates the problem. It uses the prompt method, which you used in Lesson 2, to collect information from the user.

```
<script language="Javascript">

<!--

var num1=prompt("Enter a number:","0");

var num2 = prompt("Enter a second number:","0");

document.write("num1 = ",num1,"<br>");

document.write("num2 = ",num2,"<br>");

document.write("num1 + num2 is equal to ",num1+num2,"<br>");

//-->

</script>
```

If you run the above code, and enter 5 for num1 and 6 for num2, you get the following output:

```
num1 = 5

num2 = 6

num1 + num2 is equal to 56
```

Hopefully you find this to be wrong! Whenever you use prompt to collect data, you need to be careful. Often it will not cause a problem since JavaScript will automatically convert the input string to a number when a numeric operation (such as '*' or '-') is applied. However, addition is a problem since the '+' operator may be interpreted as string concatenation.

In the above code, we want the computer to add those numbers together. But + can also mean string concatenation. So when the browser sees two strings joined by a plus sign (i.e. num1+num2), it carries out concatenation.

You can solve this problem by explicitly converting num1 and num2 into a numeric format. To do this you can use the **parseFloat method**. The word "parse" in English means to extract. So parseFloat extracts a float from a string. This method converts a string into a number. Then the browser sees two numbers joined by a plus sign, and carries out addition. Here is the same example as above using parseFloat:

```
<script language="Javascript">

<!--

var num1=parseFloat(prompt("Enter a number:","0"));

var num2 =parseFloat(prompt("Enter a second number:","0"));

document.write("num1 = ",num1,"<br>");

document.write("num2 = ",num2,"<br>");

document.write("num1 + num2 is equal to ",num1+num2,"<br>");

//-->

</script>
```

If you run the above code, and enter 5 for num1 and 6 for num2, you get the following output:

```
num1 = 5

num2 = 6

num1 + num2 is equal to 11
```

This code produces the correct result.

### JavaScript Methods That Convert Strings Into Numbers

JavaScript provides several functions that convert strings into numbers.

The method parseFloat(*string1*) returns *string1* as a decimal number, that is a number with a fractional portion.

If *string1* does not start with a number, parseFloat() gives an error message.

If *string1* starts with a number followed by some other characters, parseFloat() converts the number part of the string and ignores the rest.

---

### Examples

| | |
|---|---|
| **parseFloat**("235") | returns the number 235 |
| **parseFloat**("23.45") | returns the number 23.45 |
| **parseFloat**(``23.45abc") | returns 23.45 |
| **parseFloat**(``ab34.46") | results in an error |

---

The method **parseInt**(*string1*) returns *string1* as an integer.

If *string1* does not start with a number, parseInt() gives an error message.

If *string1* starts with digits followed by some other characters, parseInt() converts the number part of the string and ignores the following non-numeric part.

parseInt() always returns a whole number.

---

### Examples

| | |
|---|---|
| **parseInt**("235") | returns the number 235 |
| **parseInt**("23.45") | returns the number 23 |
| **parseInt**("23.45abc") | returns 23 |
| **parseInt**("ab354") | results in an error |

---

You can also convert strings to numbers with the method **eval**(*string1*), where the *string1* is a numeric expression in the form of a string. The eval method evaluates *string1* and returns its value.

---

### Example

```
<script language="Javascript">
<!—
a = "3 +2*5"
document.write(a,"<P>")
document.write(eval(a))
//-->
</script>
```

The first line output is: 3+2*5

The second line output is : 13

---

## In the Lab

This week in lab you will use JavaScript methods to convert user input from string format to numeric format, and then carry out calcuations using arithmetic operators and expressions.

Start 1st Page 2000 and begin a new HTML document. Save it giving it the name lesson0401.html. Following the instructions outlined in Appendix B, place your cursor between the <body> … </body> tags, and insert the script tags and hiding comments by using the Scriping menu.

Now type in ***exactly*** the following code:

```
<html>

<head>
   <title>Lesson 4: Formatting Input Data For Arithmetic</title>
</head>

<body>

<script language="Javascript">

<!--
var ageString;

var age;

age = parseFloat(prompt("Please enter your age:",""));

document.write("You are " + (age * 7) + " in dog years!");

//-->

</script>

</body>

</html>
```

The above program calculates your age in dog years. Try it out a few times to confirm that it is calculating the correct result.

### Student Modifications

- You can also calculate the age of a dog in human years. Prompt the user to enter the age of their dog. Use parseFloat to convert the input value to a numeric format and store it in a new variable dogAge. Then convert to human years with the following formula:

  ```
  dogToHumanYears = ((dogAge - 1) * 7) + 9
  ```

  and display the result.

- Do other conversions, from cat years (cats live about 20 years) to human years. Look on

the Internet for other possibilities: giant Redwood trees (500 to 700 years), Galapagos Turtles (200 years), etc.

## Key Terms and Definitions

- **parseFloat method** – JavaScript method that converts a string into a numeric decimal format, i.e. a number that has a fractional portion.
- **parseInt method** – JavaScript method that converts a string into an integer format, i.e. a format that is a whole number.
- **eval method** – JavaScript method that converts a string in the form of an expression into its numeric value.

## Lesson 4 Summary

In Lesson 4 you learned that JavaScript stores data input with the prompt method as a string. You also learned that data in string format cannot be used to carry out arithmetic. You learned that JavaScript provides several methods to convert strings into numbers. They are parseFloat, which converts a string to a decimal number, parseInt, which converts a string to an integer, and eval, which converts an expression in the form of a string into a numeric value.

## Lesson 4 Exercises

4_1. Sales Tax Calculator:

Write a JavaScript program that calculates the sales tax and total price of an item. Assume a sales tax rate of 8%, which means you will multiply the cost of the item by .08 in order to calculate the tax amount.

Use prompt and parseFloat to ask the user to input the item's cost and convert it to a numeric format. Declare a variable salesTax and use the formula above to calculate the sales tax amount. Declare a variable totalCost, add the item's cost plus the sales tax amount, and store it in totalCost. Output the Item cost, tax amount, and total cost on separate lines in the form of a receipt.

**Enhancements**:

- Don't assume a tax rate of 8%. Instead, ask the user to input the tax rate in the form of a decimal, i.e. .05, .03, etc., and use this tax rate for the calculations.
- Display the tax amount in a different color.
- Depending on the price of an item, you may end up with a tax amount that takes up more than 2 decimal places. Use the following code to round the tax amount to two decimal places:

```
var taxAmtRnd = Math.round(taxAmount*100);

taxAmtRnd = taxAmtRnd/100;
```

  This code multiplies the tax amount by 100, rounds it off, then divides by 100. This drops any extra decimal places from the tax amount. The rounded tax amount is in the variable taxAmtRnd.

4_2. Miles per gallon Calculator:

Write a JavaScript program that calculates miles per gallon. Use parseFloat and prompt to ask the user to input the total number of miles driven and to store that number in numeric format. Use parseFloat and prompt to ask the user to enter the number of gallons consumed. Calculate the miles per gallon with the following formula:

milesPerGallon = milesDriven/gallonsConsumed

Display the answer using the alert method.

4_3. Write a JavaScript program to assist a cashier in determining the total value of coins in a coin tray. Using parseInt and prompt, ask the user to input the number of quarters, dimes, nickels, and pennies and to store the input in separate variables. Set the default entry equal to "0", so the user can hit enter and skip to the next prompt if they do not have any of a particular coin. Then add up the total value by multiplying the number of quarters by .25, the number of dimes by .1, the number of nickels by .05, and the number of pennies by .01. Display the final total using the alert method.