Data Types & Operations

Lesson Objectives

- Understand datatypes
- Understand arithmetic, relational and logical operators.
- Understand implicit and explicit type conversions.
- Learn to use Math object.

Data Types

- All programing languages have idea of data types, some languages are strict on data types, and others are loose.
- In JavaScript, variables gets it type based on the content it is currently referring to.
- When a variable is not assigned to a value, by default its value is `undefined`, which is a special primitive type. Other primitive types are:

Type	Definition	Example
boolean	stores true or false	`let new = true;`
number	an integer or fractional number	`let count = 10;` `let average = 5.25;`
string	single or sequence of characters	`let alpha = 'a'` `let city = "Fairfield"`

- function, array, object are other complex types.
- `typeof` operator can be used to find the datatype during runtime.

Example

```
let x = 5;
let y = 7;
let z = x;
x = "Hi" ;
y = z;
z = false;
console.log(x); // Hi
console.log(typeof x) // string
console.log(y); // 5
console.log(typeof y); // number
console.log(z); // false
console.log(typeof z); //boolean
```

Operations & Operators

- Different set of operations can be performed on a variable based on its data type
  - Arithmetic operations using operators (+, -, *, /, %, ++, --)
    - When at least one of the operand type is string, + operator will perform string concatenation.
  - Comparisons using relational operators (==, !=, >, <, >=, <=)
  - Logical operations using operators (&&, ||, !)

String concatenation using + operator

- In computer programming, a string is traditionally a sequence of characters, either as literal constant or variables.
  - `alert ("this is string literal constant");`
  - `let name = "Rakesh Shrestha";`
- In JavaScript, either single or double quotes can be used to surround text to make string literals.
- Strings can be concatenated using + operator to form a long string.
  - "Hello! " + name + " how are you?"

Arithmetic Operations

- These are similar operation as we learn in mathematics (algebra)
  - With same precedence
- The modulus operator (%) gives the remainder whereas the division (/) operator gives back the quotient.
- Multiplication (*) operator is no more implicit.
- What will the result of following arithmetic operation?
  - 2-9+8-6+5

Converting a string to a number

- User input always comes as string, even when user may have entered a number.
- Before arithmetic operation can be performed, string should be converted to numeric type.
  - `parseInt(string)`
  - `parseFloat(string)`

- See example: lesson2/parsing_user_input.js

Implicit type conversion

- In most cases JavaScript automatically performs type conversion/ casting / coercion

- See example: lecture_codes/lesson2/implicit_type_conversion.js

Exercise

- Following program ask user to input temperature value in degree Celsius and outputs the result in degree Fahrenheit. Make this program running in your machine.

```
let prompt = require('prompt-sync')();
let tempInCelsius = prompt('Eneter value in celsius: ')
let tempInFahrenheit = 9/5*parseFloat(tempInCelsius)+32;
console.log(tempInFahrenheit);
```

- Now write a program that ask user to input temperature value in degree Fahrenheit and outputs the result in degree Celsius.

More assignment operators

- These are merely shortcuts

Operator	Example	Equivalent
+=	x += 2	x = x+2

- Same rule for -, *, / and % operators

Increment and Decrement Operators

- Shortcuts to increment and decrement current value by 1

  ++ count; => count += 1; => count = count + 1;

  -- count; => count -= 1; => count = count – 1;

- Pre vs Post, increment and decrement

  ++ count vs count ++

  -- count vs count –

- See example: lecture_codes/lesson2/increment_decrrement_opeartos.js

Operator Precedence Revisited

Operator(s)	Name(s)
`()`	parentheses
`- (unary)`	negation
`++  --`	increment, decrement
`*  /  %`	multiplication, division, modulus
`+  -`	additions, subtraction
`=  +=  -=  *=  /=  %=`	assignments

Math Object

- Inbuilt object as a part of JavaScript language, that provides functionality necessary for mathematical computations.
- See example: lecture_codes/lesson2/math.js

Exercise

- Write a program that computes volume of a cylinder based on user inputs for radius and height of a cylinder, using formula $v = \pi r^2 h$

- Write a program that takes x and y co-ordinates of two points as inputs and computes distance between these two points based on the formula, $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Relational operators

- As the name implies, these operator compares two values and results to either true or false (Boolean values).
- <, >,<=, >=, ==, ===, !=, !==

- See example: lecture_codes/lesson2/relational_operators.js

Logical Operators

- Logical operators (&&, ||, !) also results into Boolean values, but logical operators are typically used with Boolean (logical) values.
  - Recall, relational operators results into Boolean values.
  - So, logical operators can be to used to combine two or more relational expressions.

  - See example: lecture_codes/lesson2/logical_operators.js

Truth Tables

Exp1	Exp2	Exp1 && Exp2	Exp1 || Exp2
F	F	F	F
F	T	F	T
T	F	F	T
T	T	T	T

Exp	!Exp
F	T
T	F

Main point

- Data type determines the operations that can be carried out on data. Same operator can do different operation depending on the data type. *Science of consciousness, Experiencing the pure field of consciousness is one operation that every human nervous system is capable of. TM operates in same way for every human nervous system, when practiced correctly.*

Truthy & Falsy

- In JavaScript, any value can be used as a Boolean
  - "**falsy**" values: 0, 0.0, NaN, "", null, and undefined
  - "**truthy**" values: anything else

Short-circuit evaluation

- The && and || operators actually return the value of one of the specified operand, so if these operators are used with non-Boolean values, they will return non-Boolean value.
    - See example: lecture_codes/lesson2/short_circuit_or.js

Assignments

- Readings
  - https://www.w3schools.com/js/js_datatypes.asp
  - https://www.w3schools.com/js/js_operators.asp

- Programming Assignments from Chapter 5
  - Questions 1 to 7

- **Note, you won't be using HTML form input/output as used in book. Instead you used console input/output in NodeJS**.
- Upload your assignments to your GitHub repository and submit your GitHub link on Sakai.