Topics:

  DataTypes:
    Primitive:
      number, string, boolean, null, undefined, bigint, symbol (for unique key)

      typeof - return the string ('number', 'boolean', 'string', 'object'...)
      typeof typeof - 'string'

    Composite: combination of single/multiple types
      Array --> [1, 'sanjay', boolean, null]
        Note: an array can also contain compsoite types (Array, Object)

          Eg: Array can contain array itself

            [[1, 2, 3], ['Sanjay', 'Santosh', 'Manoj'], ['Python', 'JavaScript']]

              Array can also contain objects
            [
              { name: 'Manoj', age: 12, origin: 'Tanjavur' },
              { name: 'Sanjay', age: 25, origin: 'Chennai' },
              { name: 'Manikandan', age: 24, origin: 'Bengaluru' }
            ]

      Object --> {
        name: 'Sanjay',
        role: 'Developer',
        expe: 10,
      }
        Object Value can also be a composite value
          Eg: Object can contain array, object
              {
                "skills": ["Python", "Javascript"],
                "contact": {
                  "mobile": "9090909090",
                  "email": "sanjay@gmail.com",
                  "website": "sanjay.com"
                },
                "isIndian": true
              }

  Functions:

    used to create a functinality using set of statements with optional return statement

    normal function
    arrow function

  {} - object elements
  [] ---> Array elements / array elements accessing
  () ---> methods/functions

  Scopes:
    global ---> parent scope every scopes ()

      function/local scope (inside function (){})

      block {}

    {} ---> block of lines

    var - global/function level scope
    let, const - global, function, block (Lives only the scope)

Task:
  Practice all the data declaration & initialization using different scopes
  Visualize it in Python Tutor: https://pythontutor.com/visualize.html#mode=edit

Next Session:
  difference between normal function & arrow functions

  Operators:

    Arithemetic: +, -, *, /, %(5 % 2 ---> 1 (remainder))
    Assignment: =
    Logical: !(not), &&, ||
    compariason: >, <, >=, <=, !=, ==, ===
    Boolean: true or false

  Conditional Statements:
    if else
    switch case
    == vs ===

  looping, conditions, simple problems

    looping: it is a process of repeating a block statements again and again

    for, while, do while

    how many times or when to stop the looping

    concepts in looping
    initialization, termination condition, inc/drec