General(Languages)

Classification:

- abstraction
  - declarative
    - functional
  - imperative
    - procedural
- behavior
  - dynamic
    - scripting
  - static

Declarative Languages:

- not imperative
- describes what computation should be performed and not how to compute it
- lacks side effects (referentially transparent)
  - an expression always evaluates to the same result in any context
  - instance can be replaced with its corresponding value without changing the program's behavior
- clear correspondence to mathematical logic

Functional Languages:

- Def
  - define programs and subroutines as mathematical functions
  - many functional languages are "impure", containing imperative features
  - many functional languages are tied to mathematical calculation tools
  - declarative: programming is done with expressions or declarations instead of statements
  - the output value of a function depends only on the arguments that are input to the function

    - calling a function f twice with the same value for an argument x will produce the same result f(x) each time
    - eliminating side effects(changes in state that do not depend on the function inputs) makes it much easier to understand and predict the behavior of a program

- Pure ex: Haskell
- Impure ex: SML

Imperative Languages:

- def
    - uses statements that change a program's state

Scripting Languages:

- def
    - dynamic high-level general-purpose interpreted languages,
- Perl,[2] PowerShell, Python,

Hierarchy of programming languages:

- High-Level language (C, Java, PHP, Python)
    - more complex than machine code
- Assembly language
    - machine code with names substituted for numbers
- Machine code
    - only numbers
- Hardware
- convert program into machine language
    - compile the program
    - interpret the program

Web Development Languages:

- HTML/XHTML
  - Defines content of web page
- CSS
  - appearance of web page
- HTML+CSS can create static web pages
  - static pages can interact with your visitor through the use of forms
  - form is fill, request submitted and sent back to the server, new static web page is constructed and downloaded into the browser
  - disadavantage of static web pages, only way visitor interact with the page is by filling out the form and waiting for a new page to load
- javascript
  - behavior of web page
  - can validate each of the fields as visitor enter and provide immediate feedback when they make a typo (vs after they filled everything and submitted)
  - add animations into the page which either attract attention to a specific part of the page or which make the page easier to use
  - provide responses within the web page to various actions that your visitor takes
  - load new images, objects, or scripts into the web page without needing to reload the entire page
  - pass requests back to the server and handle responses from the server without the need for loading new pages
  - not everyone visiting your page will have JavaScript and so your page will still need to work for those who don't have JavaScript

sources:

- `https://en.wikipedia.org`
- `https://wiki.haskell.org/Referential_transparency`