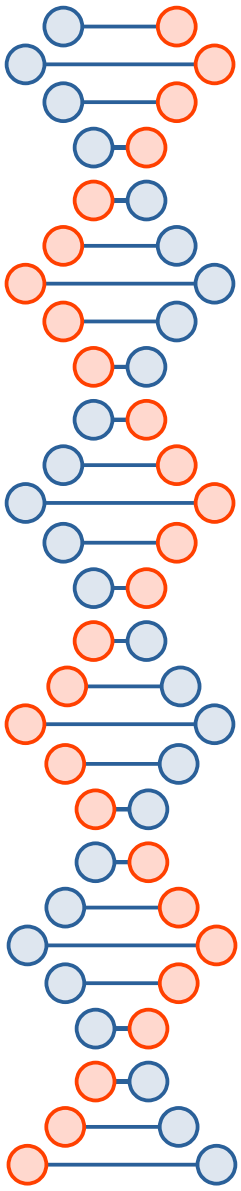
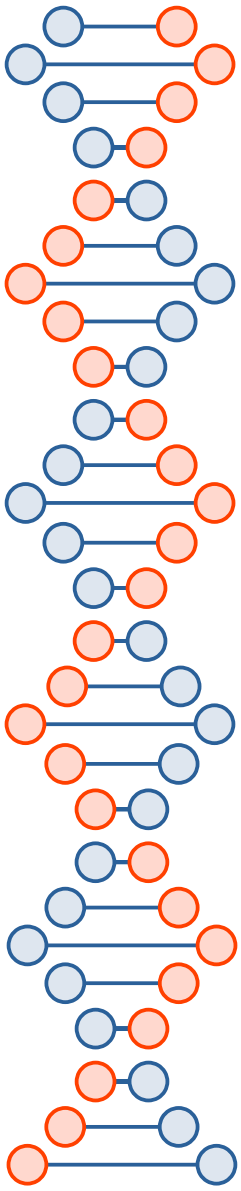


Basic Computer Science and Programming for Biologists

Dr. Haley Sanderson
Post Doctoral Research Fellow
Vaccine and Infectious Disease Organization
University of Saskatchewan
omx419@usask.ca



Outline of Instruction

- Lectures
  - 2 lectures on CS and programming basics
- Hands-on Workshops
  - Linux Command-Line
    - 3 x 90min hands-on
  - Python
    - 5 x 90min hands-on
  - R
    - 1 x 90min hands-on

Programs you need installed for Workshops:

1) Linux Subsystem for Windows
   (Marc is needed to install)
2) R
3) R Studio

A little about me!

- Post Doc in Dr. Aaron White's Group (2021 to Present)
- 5 years of bioinformatics and programming experience
- Previous genomics Post Doc with Dr. Rob Beiko at Dalhousie University (2019-2021)
- First Post Doc was in Molecular Biology with Dr. Tim McAllister at AAFC (2018-2019)
- PhD in Environmental Studies at Queen's University (2018)
- Bsc in Molecular Biology and Genetics at University of Guelph (2013)

My guinea pigs: Charlie (Left; black and blonde) and Henry (Right; Blonde and White)

Lecture Outline

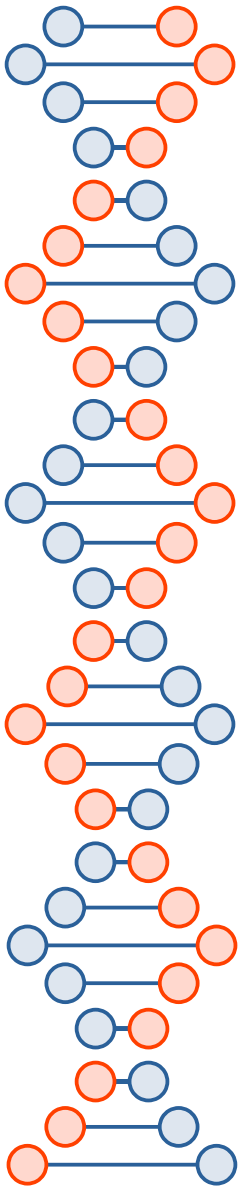
1)Computing Systems

2)Data and Analysis: Storing, Collecting and Using Information

3)Basics of Software Engineering

4)Basics of Algorithms and Programming

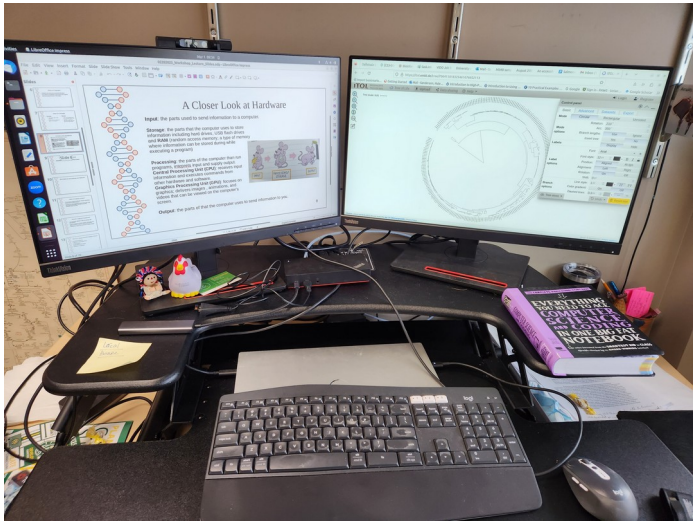
5)Universal Programming Principles



1) Computing Systems

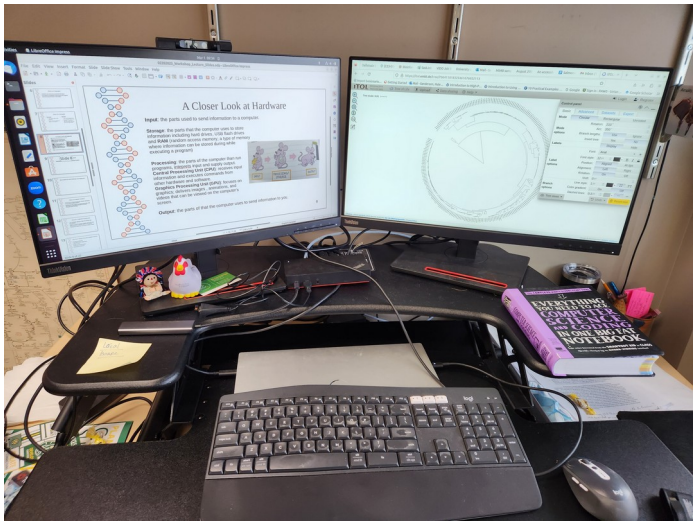
What is a Computer?

- A device that stores and processes information
- Can perform complicated computations and organize and store large amounts of information



Parts of a Computer

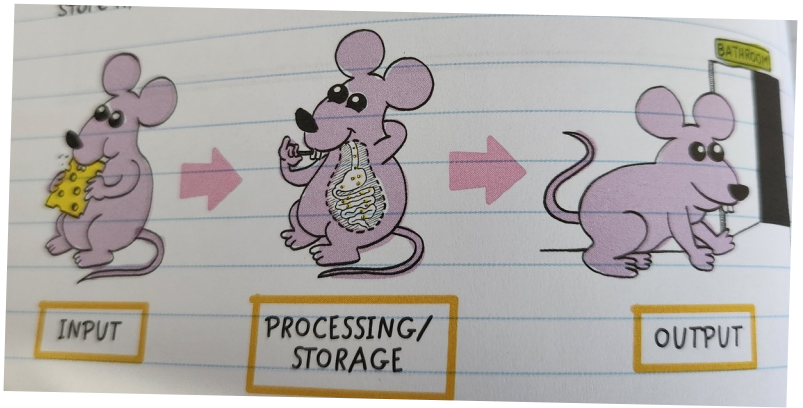
- Hardware
  - The physical part of the computer
  - If you can see it with your eyes, it is hardware.
- Software
  - The set of programs that tell a computer what to do
  - Software is stored on storage devices like hard drives and flash drives



A Closer Look at Hardware

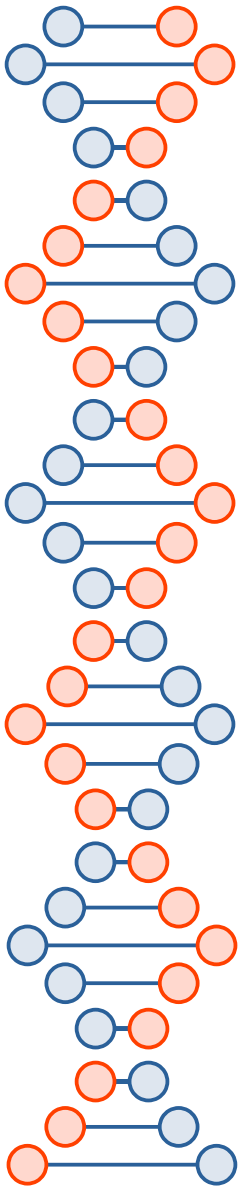
Input: the parts used to send information to a computer.

Storage: the parts that the computer uses to store information including hard drives, USB flash drives and **RAM** (random access memory; a type of memory where information can be stored while executing a program)



Processing: the parts of the computer than run programs, interprets input and supply output.
Central Processing Unit (CPU): receives input information and executes commands from other hardware and software.
Graphics Processing Unit (GPU): focuses on graphics; delivers images , animations, and videos that can be viewed on the computer's screen.

Output: the parts of that the computer uses to send information to you.

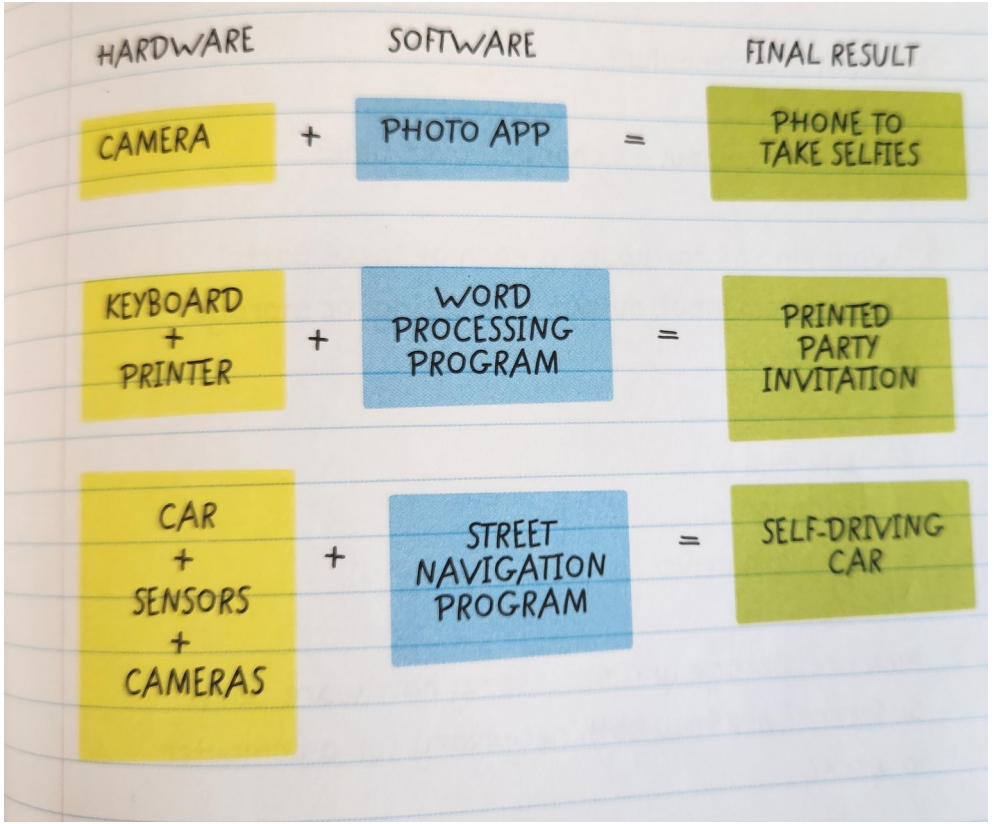


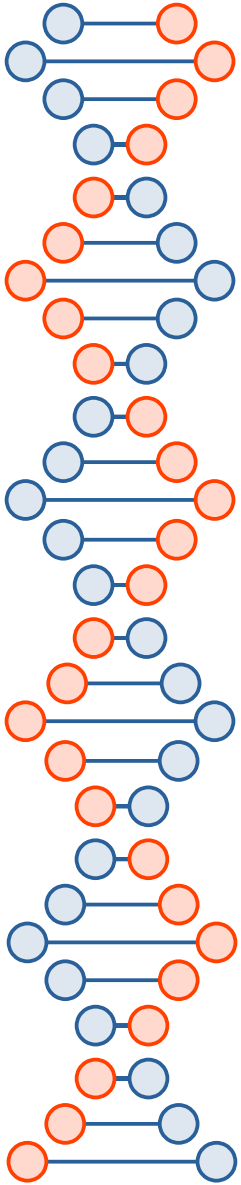
A Closer Look at Software

- Applications
  - Programs that allow the user to complete tasks
  - Examples: word processors, internet browsers and games
- Operating Systems
  - Programs that make sure that individual hardware devices work together and properly with other programs
  - Examples: Windows, Linux

Applications cannot work without the system software.

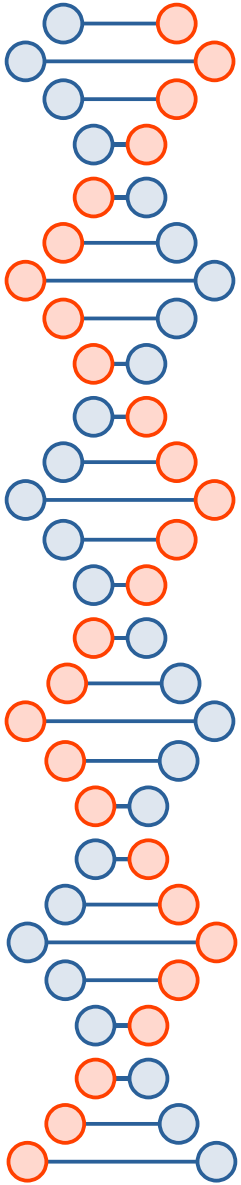
Both Hardware and Software are Necessary





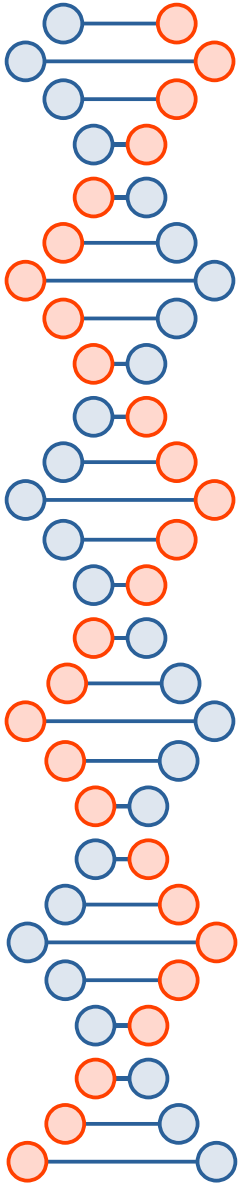
What is Computer Science?

- The study of computers and how computer technology can be used to solve problems
- Studying computing systems, programming rules, data and analysis, networking, the internet and how computers affect our lives



What do Computer Scientists do?

- Use computers' abilities to help them develop new technologies
- Program computers to both complete tasks better and faster and teach computers how to do new things
- Shifting from being only a consumer to being a creator

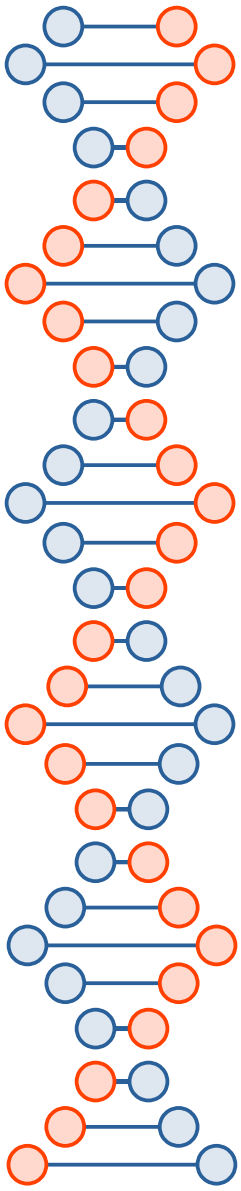


The Five Concept Areas of Computer Science

1) Computing Systems
2) Algorithms and Programming
3) Data Analysis
4) Networks and the Internet
5) Impacts of Computing

What is a Computer System?

- All the basic hardware and software that work together to make a computer run
- Examples: desktop computers and laptops, smartwatches, phones
- Many electronic devices are run by programs and have built-in computers

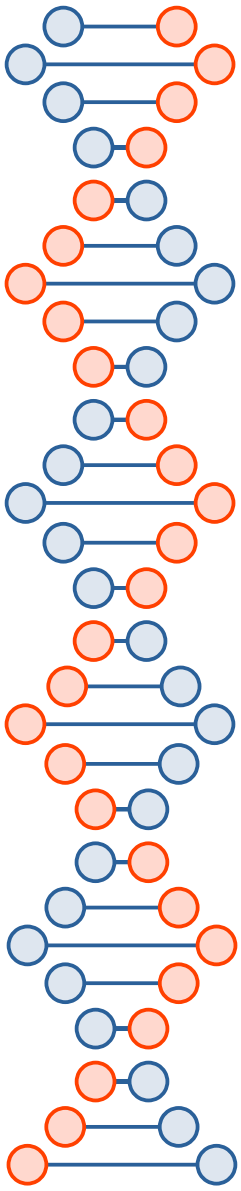


Algorithms and Programming

- Involves writing programs that tell computers what to do
- Programs can be very complex or very simple
- Writing a Program
  - Step 1: make a list of step-by-step instructions for what they want their program to do
  - Step 2: translate instructions into a language (code) the computer can understand. The algorithm is now a program.

Algorithm: a list of steps or instructions written in human language that tells a person how to complete a task

Program: a set of instructions (or an algorithm) that has been translated into commands a computer can understand (code)



Data and Analysis

Data: raw, unorganized facts

Analysis: organizing, describing and understanding data

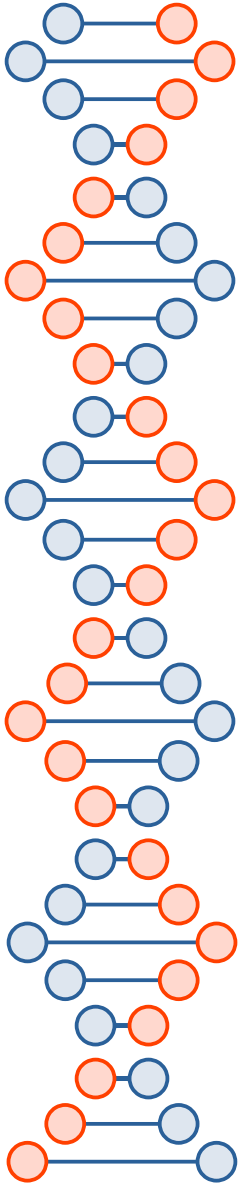
Computers are great at storing, sharing and calculating large amounts of data and are great for analyzing data.

Networks and the Internet

- Networks: are a group of connected devices. They share information and resources
- Internet: the worldwide network that connects millions of computers

Impacts of Computing

- Computers influence our lives, culture, safety, laws and behaviour
- There are ethical considerations for every program and computer system

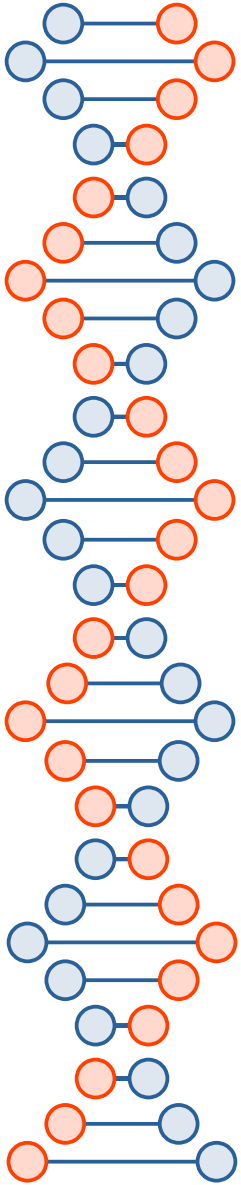


How do we interact with computers?

User Interface (UI):

includes all the parts of a computing system that you use to operate the computer

Example: playing a mobile game the UI would be the touch screen, speakers, and the game's onscreen menus, buttons and graphics



GUI vs CLI

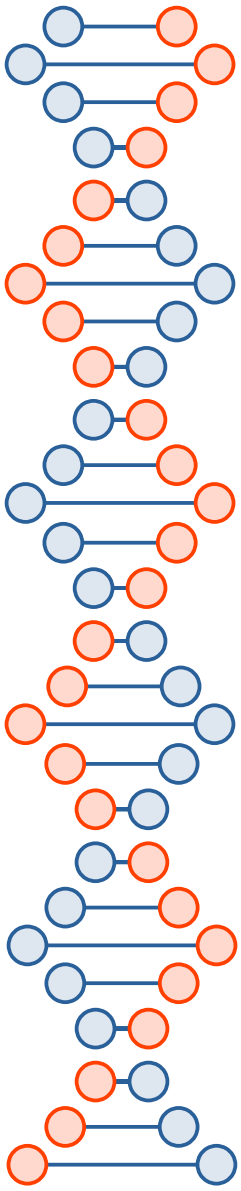
- Graphical User Interface (GUI)
  - Uses icons and symbols on the screen instead of just plain text
- Command-Line Interface (CLI)
  - Uses only text to operate the computer

Troubleshooting

Troubleshooting: taking a systematic, or step-by-step, approach to solving errors within a computing system or software.

Systemic Approach: following a fixed plan or process to solve problems

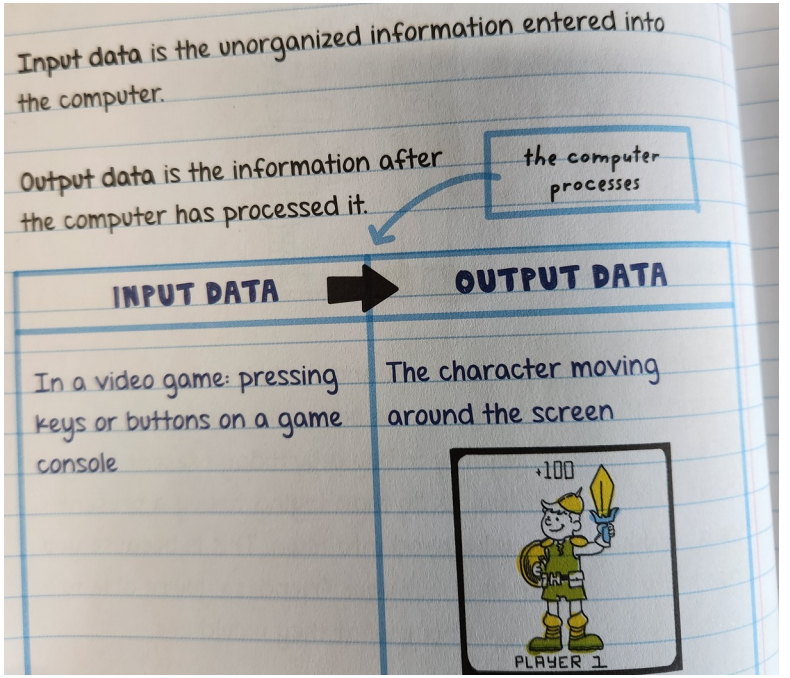
Debugging: finding and correcting errors within a program



2) Data and Analysis

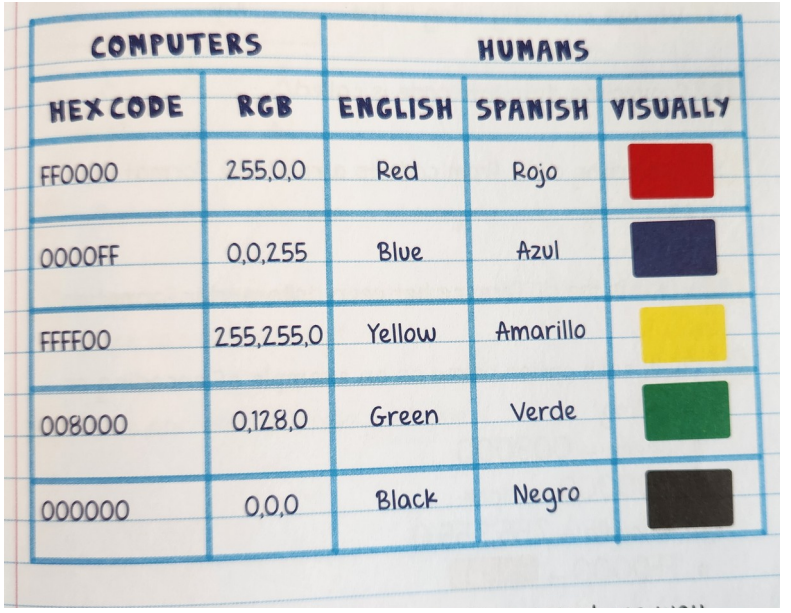
Computers Use Data

- **Data**: a collection of unorganized figures, words, and numbers that have not yet been given meaning
- **Information**: data that has been organized to have meaning and context

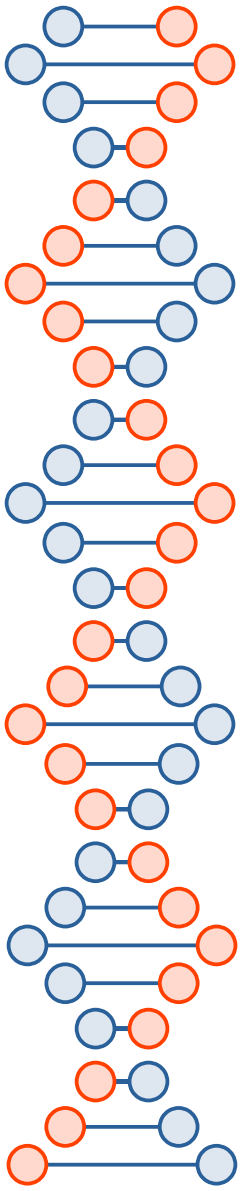


Encoding Data

- Code: a system of symbols, letters and numbers used to represent something else
- Encode: writing data in a format that the computer will understand
- Decode: convert code to an understandable form of communication
- Data Encoding Schemes
  - Binary code
  - Colours: RGB and Hexidecimal

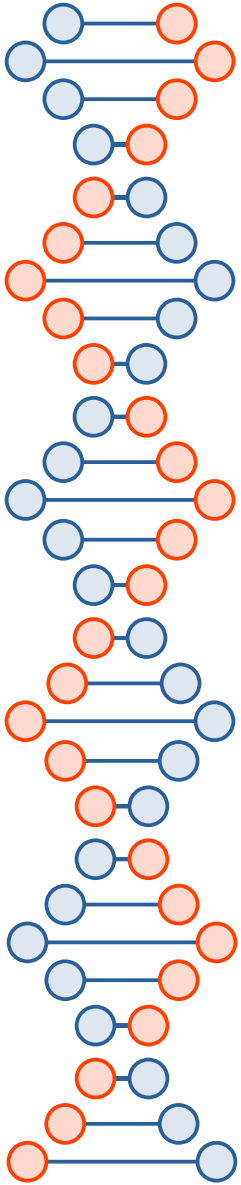


COMPUTERS		HUMANS		
HEX CODE	RGB	ENGLISH	SPANISH	VISUALLY
FF0000	255,0,0	Red	Rojo	
0000FF	0,0,255	Blue	Azul	
FFFF00	255,255,0	Yellow	Amarillo	
008000	0,128,0	Green	Verde	
000000	0,0,0	Black	Negro	



Collecting and Using Data

- Collecting
  - Interviews
  - Surveys
  - Observations
- Representing
  - Charts and graphs
    - Line graph
    - Bar graph
    - Pie charts
    - Phylogenetic trees



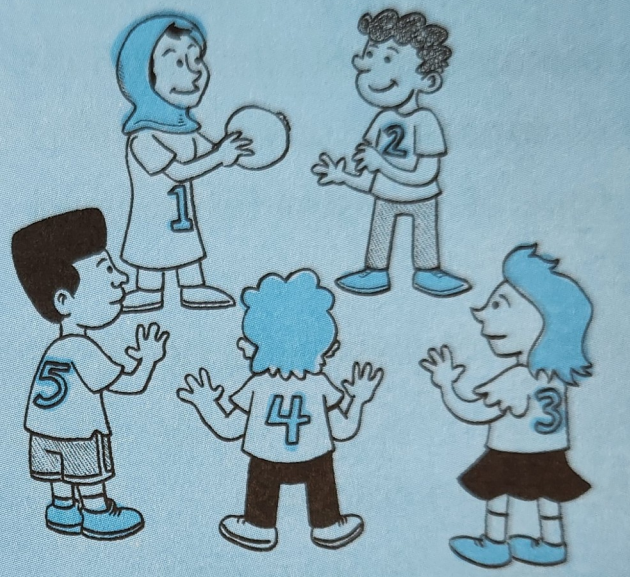
3) Basics of Software Engineering

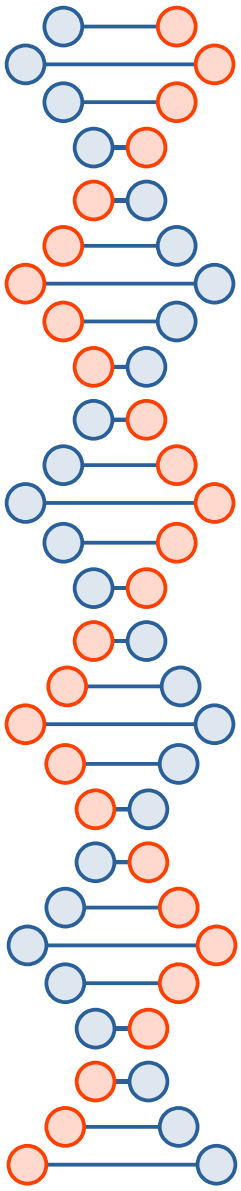
The Engineering Design Process

There are many slightly different **engineering design processes**, but a basic version is

1. identify the problem
2. plan a solution
3. build/make the plan
4. test
5. improve

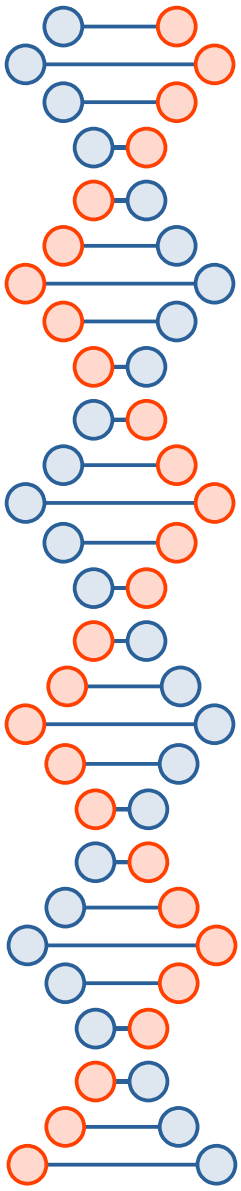
and then the cycle starts again.





Testing

- Defensive Programming: designing your program so that it keeps working even when things don’t go as expected
  - Use and Test cases are used to help you see where adjustments need to be made
- Use Cases: lists of actions users can take and describes how users will use your program
- Test Cases: specific and more detailed tests; focus on one condition (or variable) at a time to make sure the program works in all situations



Documenting

Documentation: information about the program's code.

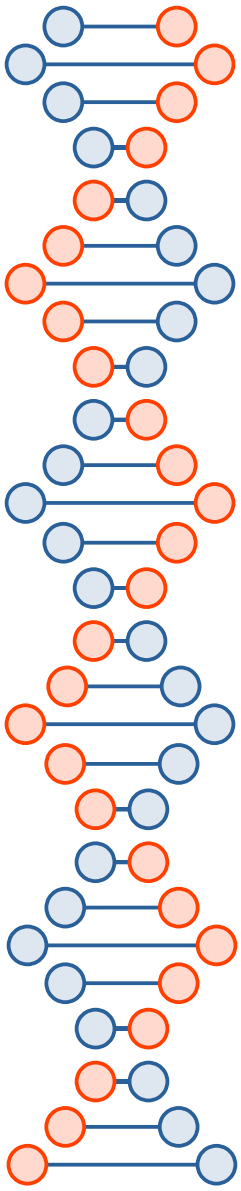
Two types: Comments and README files

Comments: messages written by the programmer about the program's code; tells you what each part of the code does

Commenting out: marking sections of code as a comment so that it will not be run with the rest of the code

README file: gives information about a program including which files are included, how to use it, credits and attributions, how to install it, etc.

Usually a Plain Text File



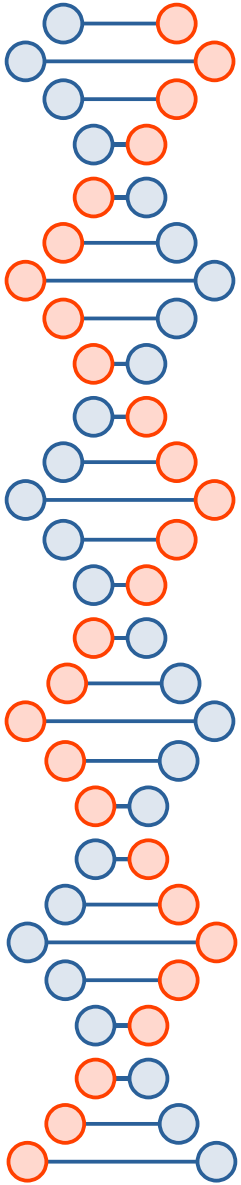
Incorporating Feedback

User-Centered Design

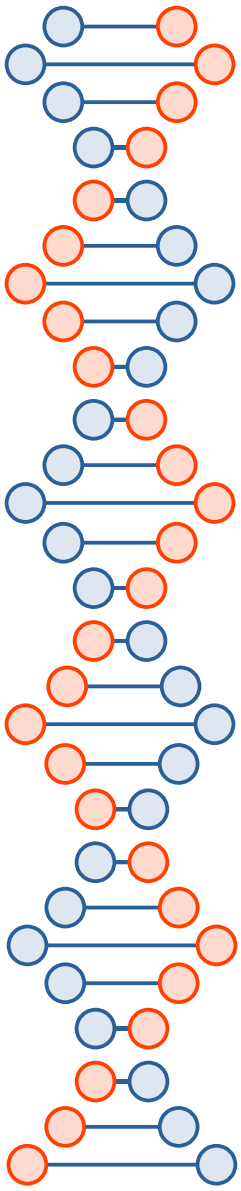
- A process for creating a program that considers users' wants and needs at every stage of development
  - Usability
  - Accessibility
  - Content

Collecting Feedback

- **Alpha Testing**: first round of user testing, which is usually done just before the program is finished.
  - Testers are usually colleagues.
- **Beta Testing**: second round of user testing, which is usually done after the program is finished.
  - Testers are usually a selected group of potential users.



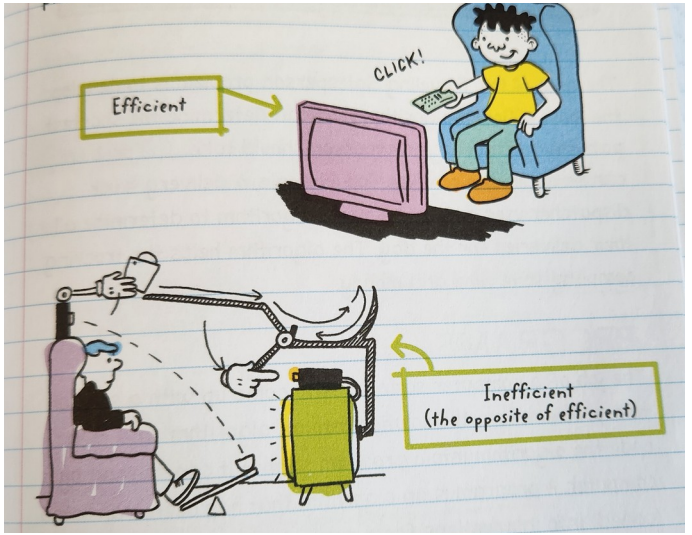
Designing programs is usually done through collaboration and it is important to give credit to those that have contribution.

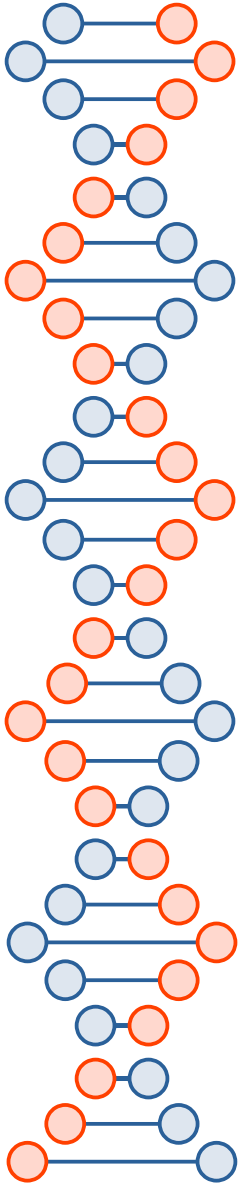


4) Basics of Algorithms and Programming

What are Algorithms?

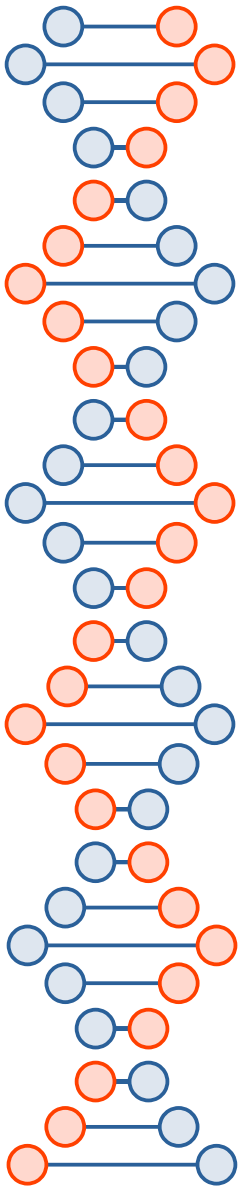
- Processes or steps to be followed that give clear instructions for repeating tasks
- Some algorithms are better than others
- The best algorithms are fast, simple and efficient





What is a Program?

- **Program**: An algorithm that has been translated (coded) into instruction for a computer
- **Programming Languages**: use a combination of numbers, words, symbols, and formatting to tell a computer what to do in a way it can understand
  - Examples: Linux/Unix, Python, R



Using Programming Languages

- Different programming languages have different strengths and weaknesses
- Most large projects use multiple languages

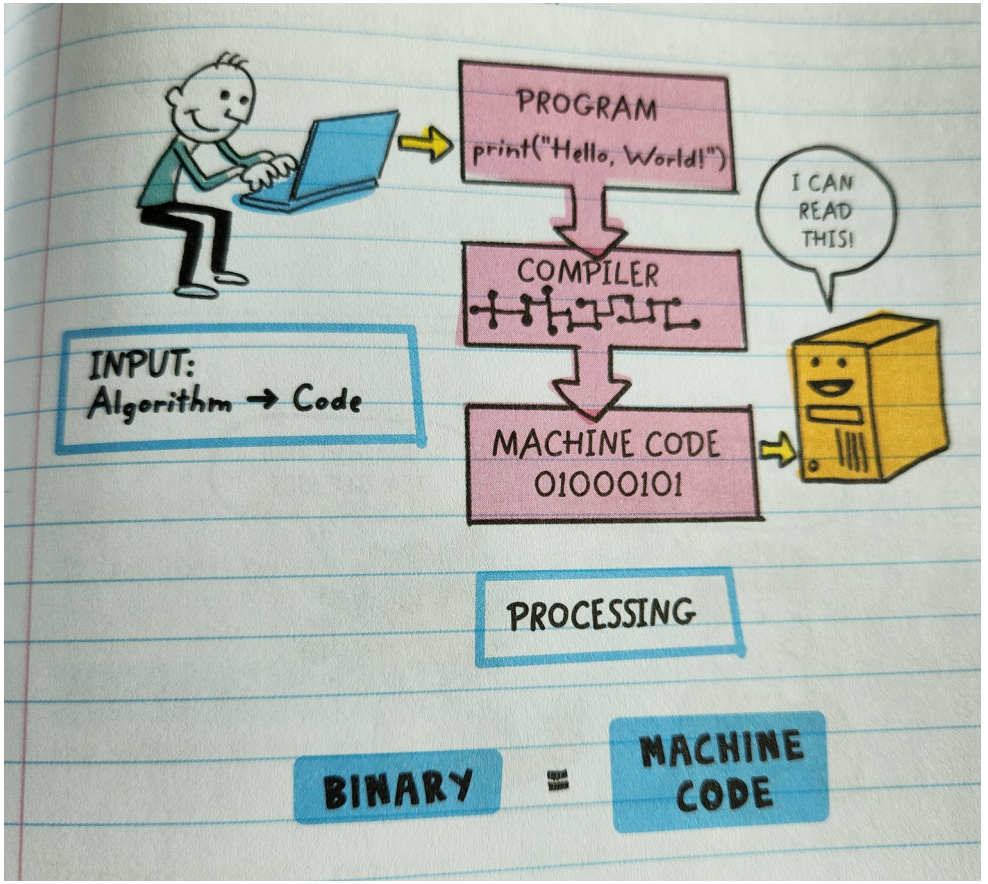
- Example: Twitter uses JavaScript, C++, Ruby and others

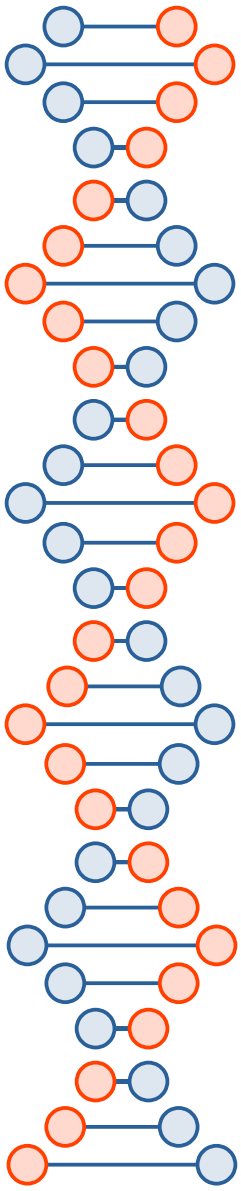
Front-End vs Back-End Languages

- Front-End: used to make the part of a website or program that can be viewed on the screen
- JavaScript

- Back-End: Organizes and connects all the information to the front end
- C++, python

How Computers Read Code



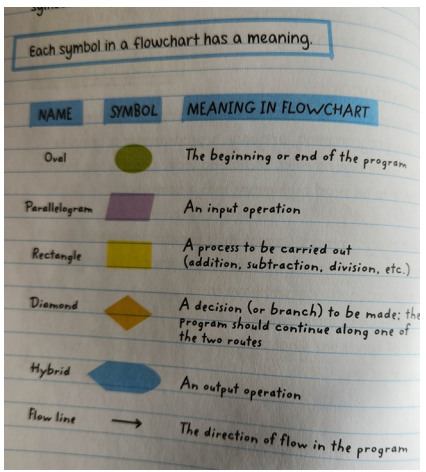


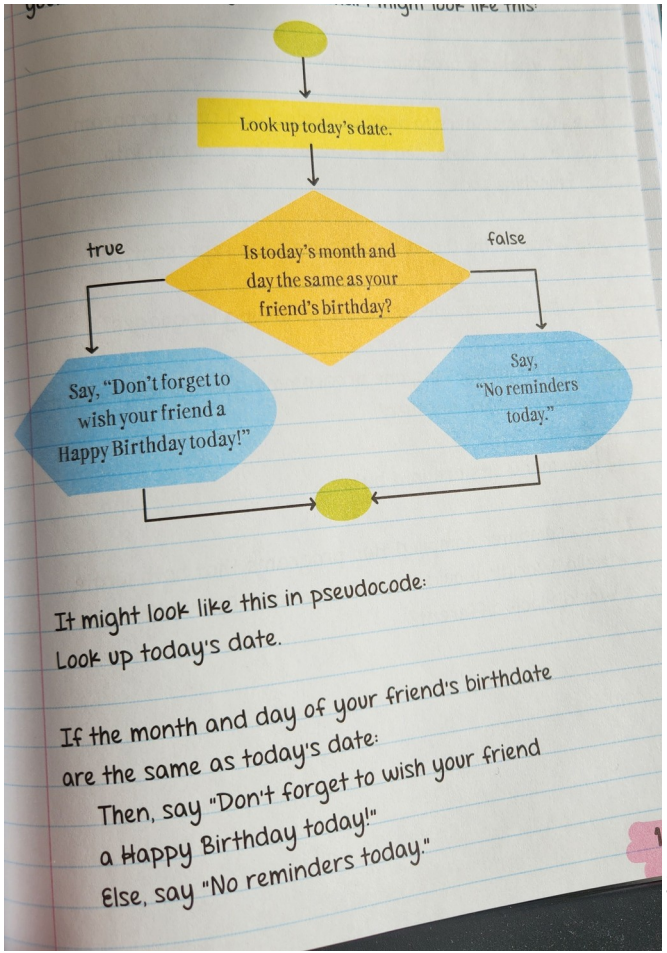
Bits and Bytes

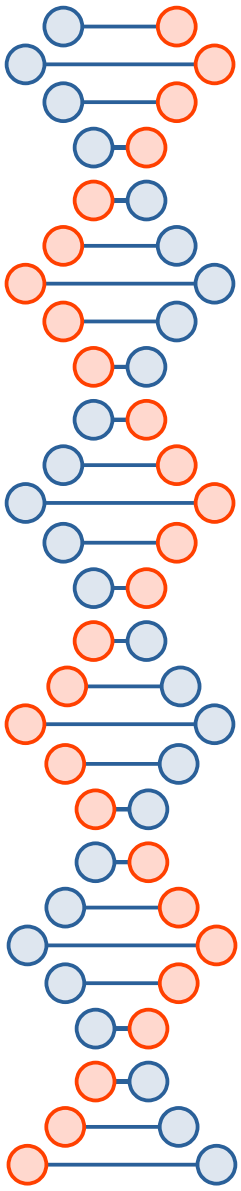
- Bit (b): the smallest unit of storage representing one digit (1 or 0)
- Byte (B): next greater measurement which represents 8 bits
- Kilobyte (KB): ~1000 bytes; word documents
- Megabyte (MB): ~1 million bytes; audio file
- Gigabyte (GB): ~ 1 billion bytes; HD movie
- Terabyte (TB): ~ 1 trillion bytes; Most external hard drives

Creating an Algorithm

- **Pseudocode**: written in a style similar to the programming language used
- **Flow Charts**: used to visualize each step of code

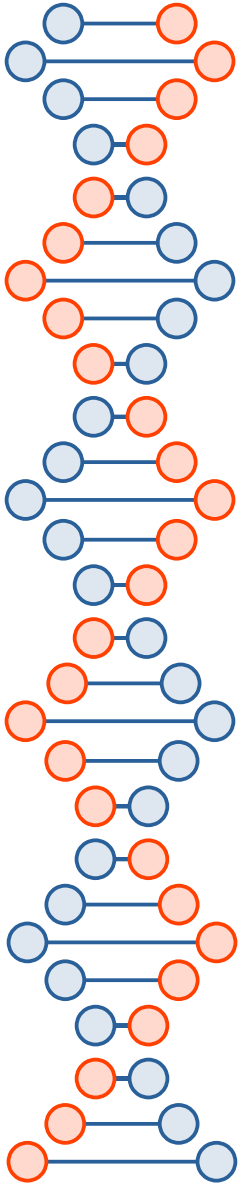




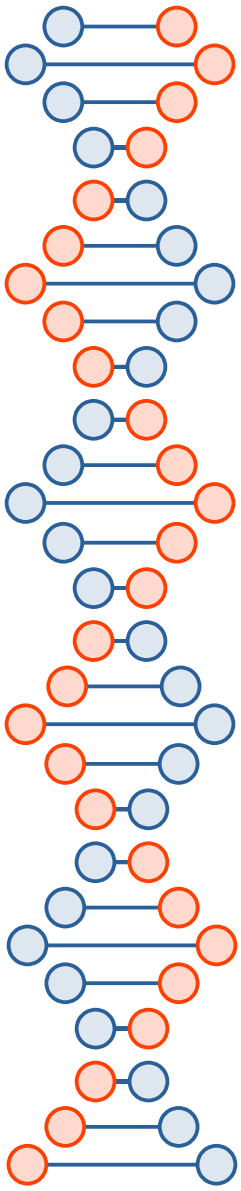


Computational Thinking

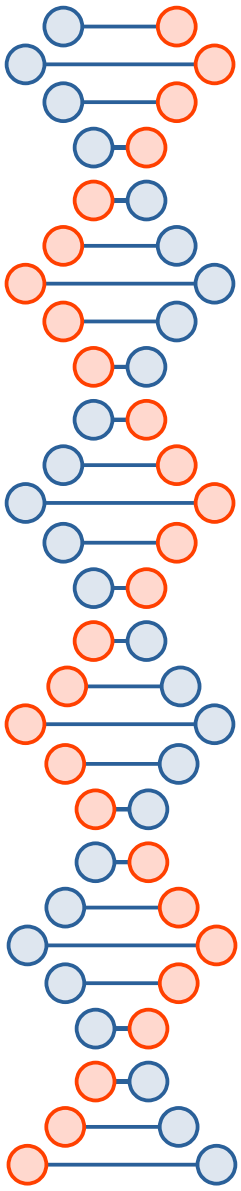
1) **Decomposition**: breaking a problem down into simple parts
2) **Pattern Recognition**: identifying what different problems have in common
3) **Abstraction**: separating details that matter from details that are not important
4) **Algorithm Design**: creating a solution with simple steps that anyone can follow



End of Lecture 1
Questions?

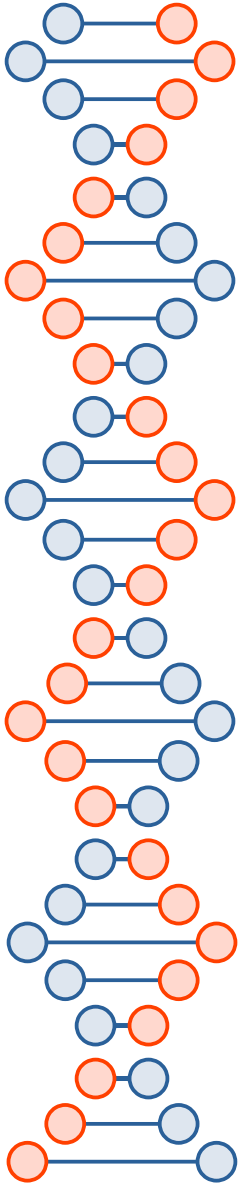


Lecture 2
5) Universal Programming Principles



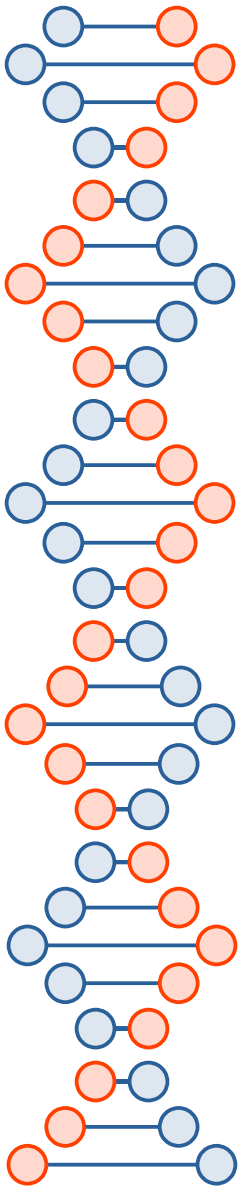
Did you get 2 emails from me?

- Yes, you are on the mailing list
- No, you are not on the mailing list --> email me at omx419@usask.ca
- You need to be on the mailing list to get workshop materials and lecture slides
- Hands-on Workshops in Linux command-line start next time (March 31st, 9-11am)



Universal Programming Principles

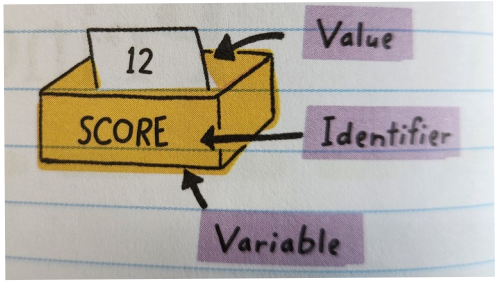
1) Variables
2) Conditional Statements
3) Loops
4) Events
5) Procedures



1) Variables

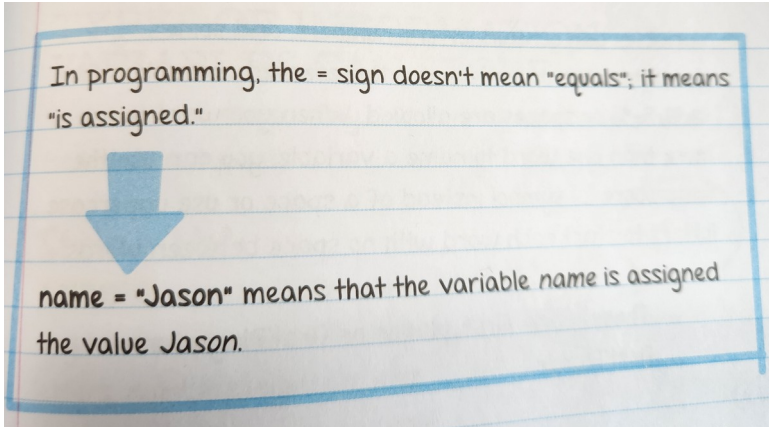
What are Variables?

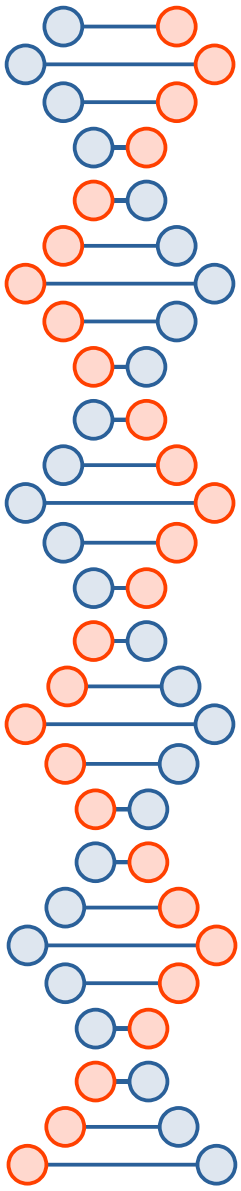
- In math, a variable is a letter or symbol used in place of a quantity that is unknown
- In Computer Science, variables act as stand-ins or placeholders for values that are stored in the computer
- **Identifier**: name of the variable
- **Value**: information that the variable contains which can be text, numbers or other types of data



Assigning and Naming Variables

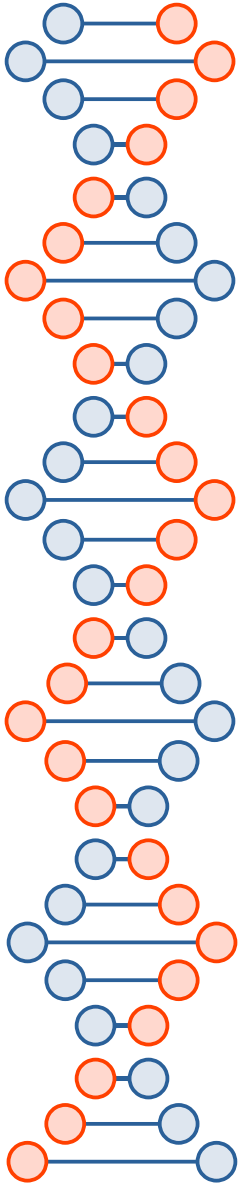
- To declare a variable is to create a variable within a program by giving it a name
- To assign a value to a variable, use the equal sign (=) which is an assignment operator
  - Left is identifier, Right is the value





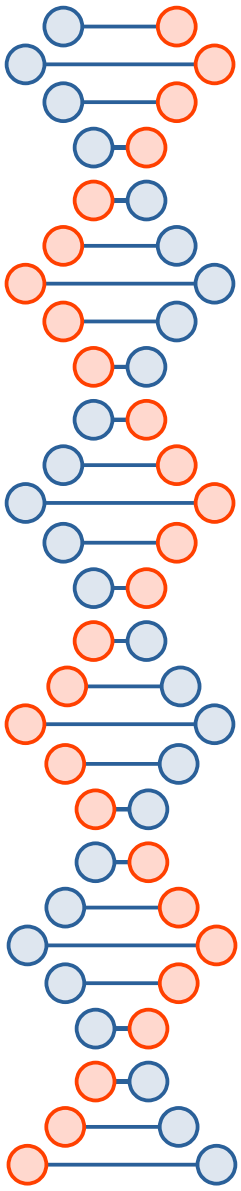
Specific Rules for Identifying and Naming Variables

1) Variable names should be short and should clearly describe what the variable represents
2) Variables should begin with a letter of the alphabet
3) Variables can contain letters, numbers, and some characters (including underscore) but no reserved characters (#,@,&,%) or words that have special meaning in the programming language

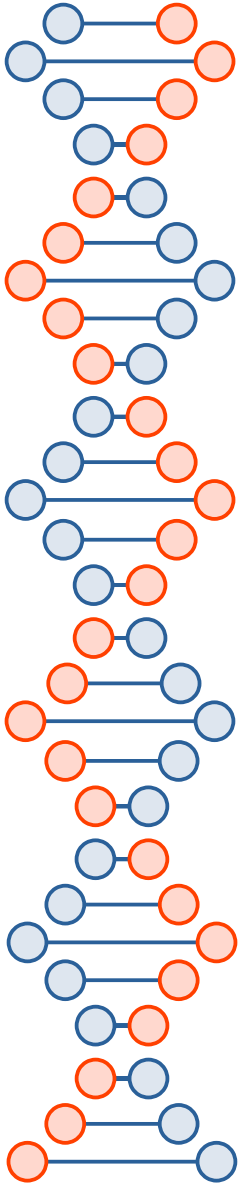


Types of Information Variables Can Store

- **String Values**: variables can store any kind of character; a string is always placed inside quotation marks
- **Numeric Values**: including integers and floating point numbers, algebraic expressions; do not place in quotation marks
- **Boolean Values**: True or False
- **Lists or Arrays**



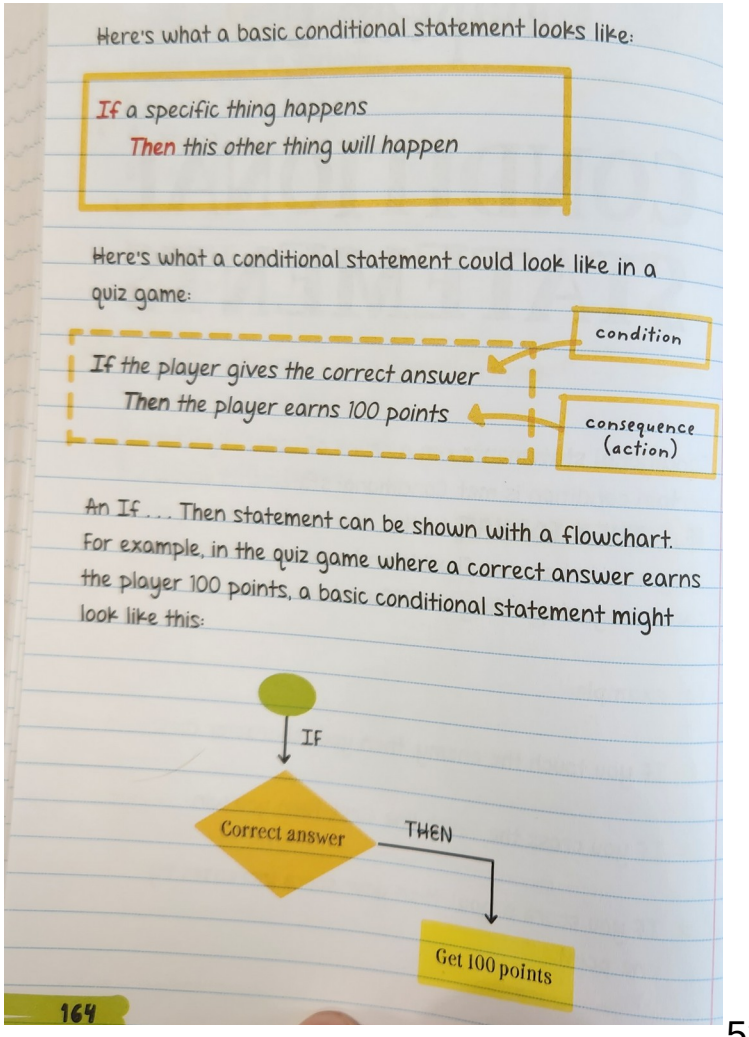
2) Conditional Statements



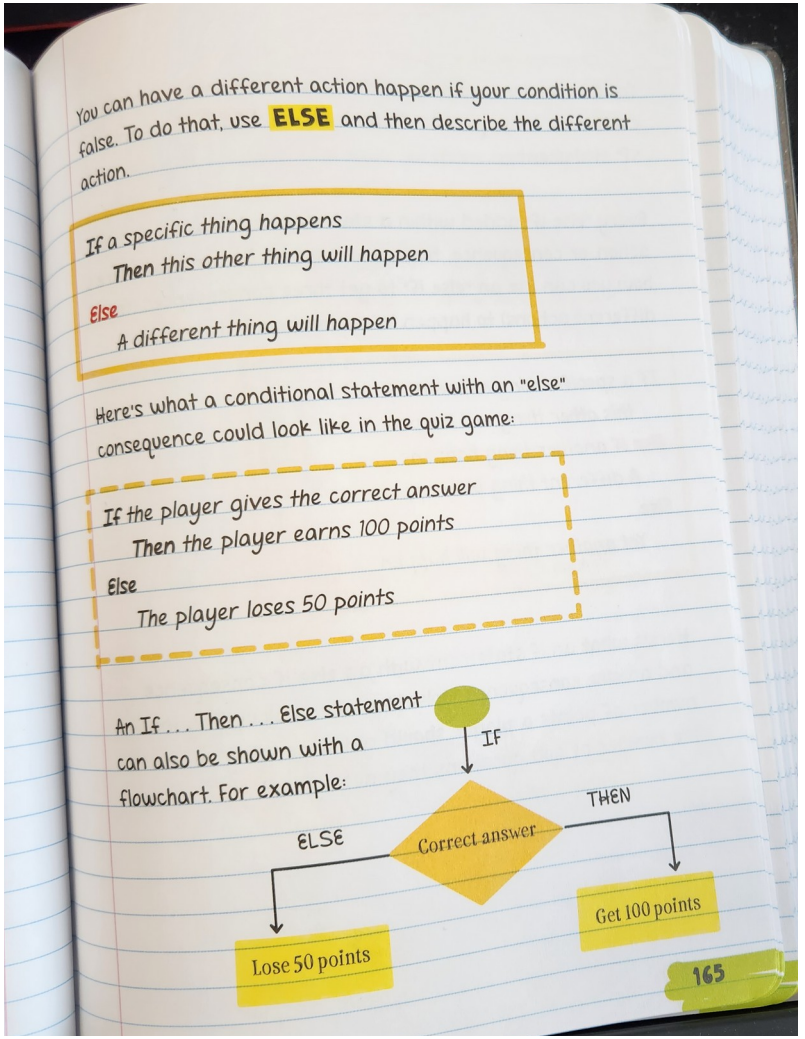
What is a Conditional Statement?

- Run a chunk of code only when a certain condition is met
- Follows the **IF...THEN** format
- Allow programs to be more flexible because they can change depending on different conditions

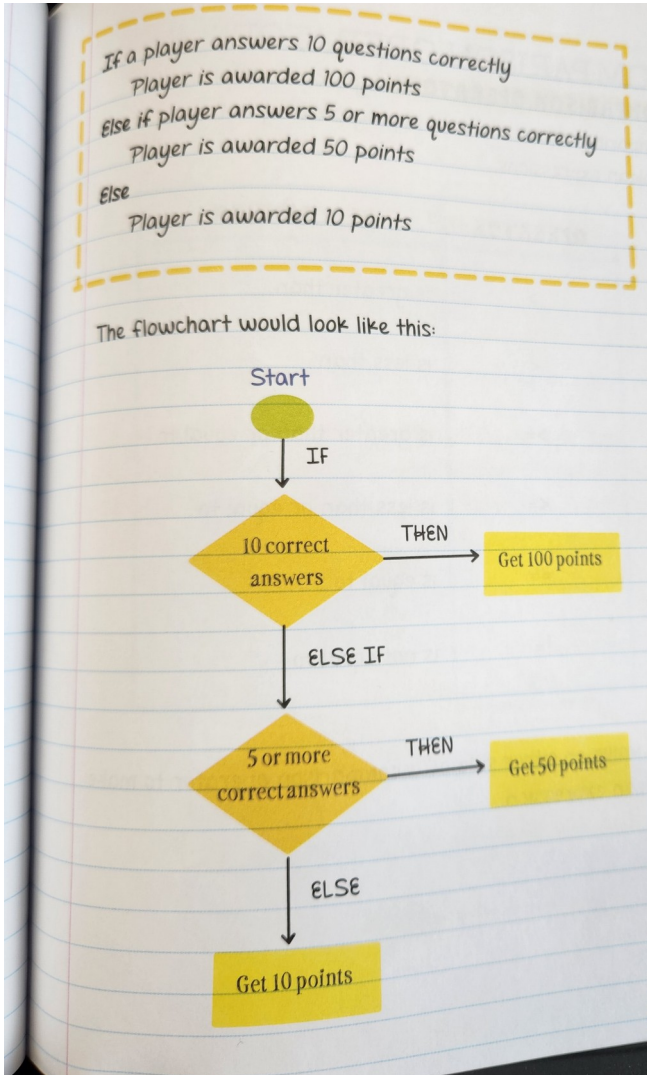
Example of Basic Conditional Statement



Example of Basic Conditional Statement: Else

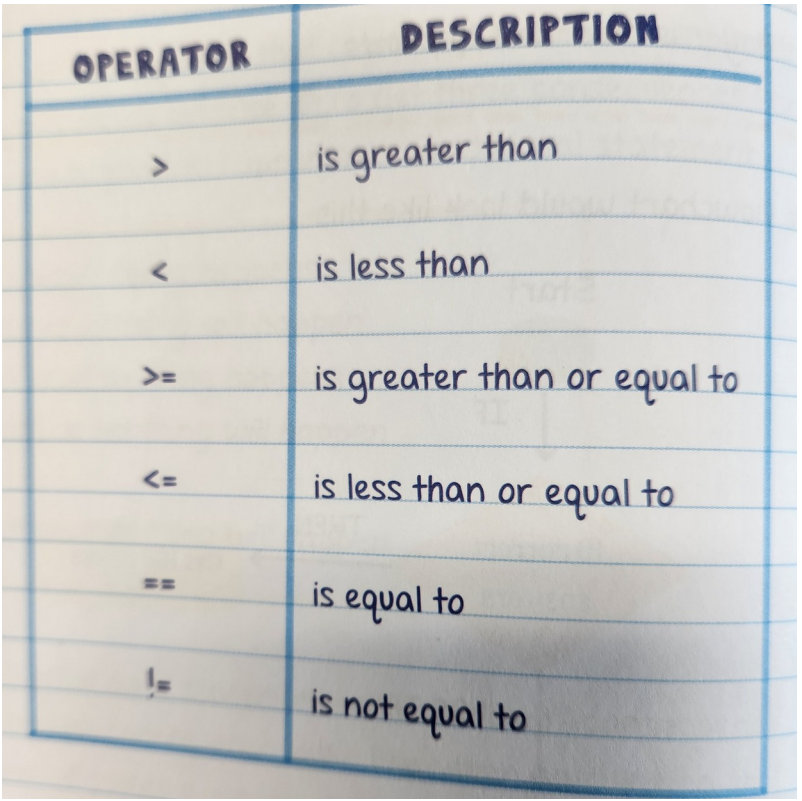


Example of Basic Conditional Statement: If Else



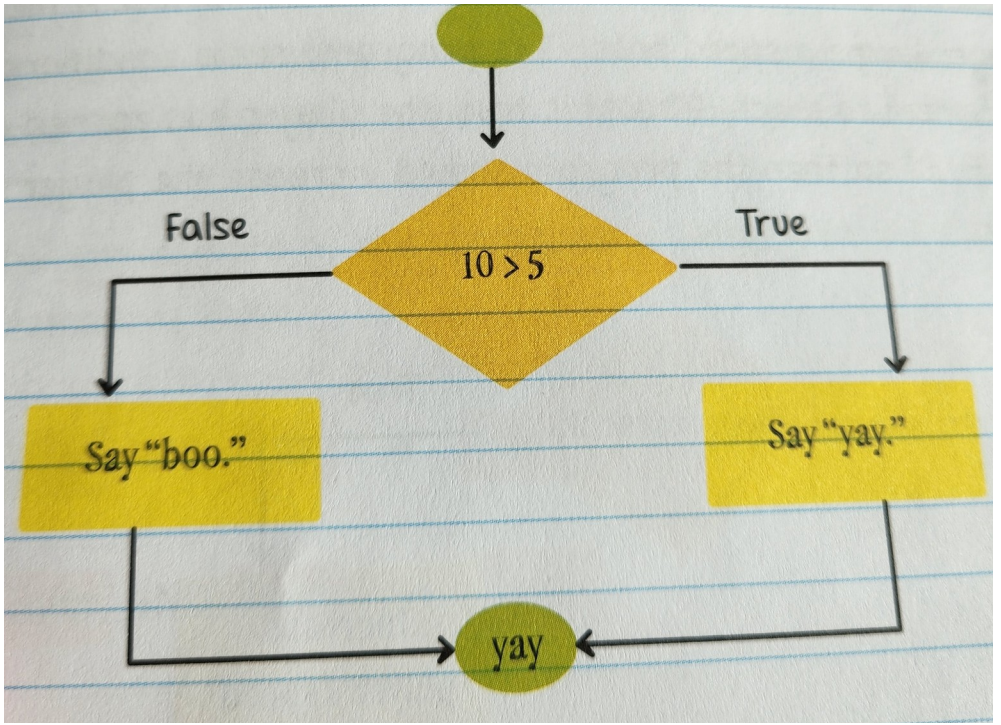
What are Comparison Operators?

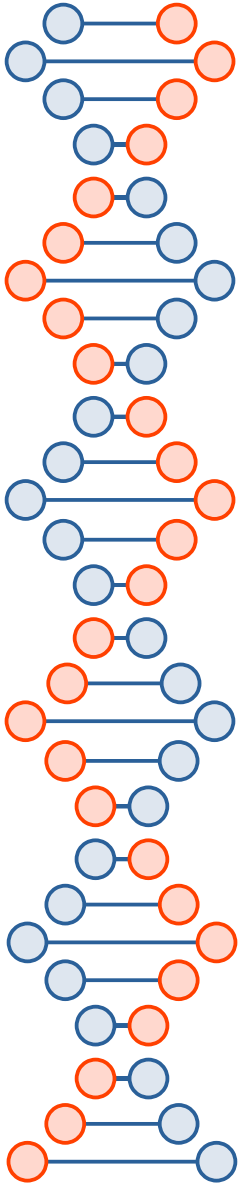
- Symbols used when comparing values
- Used in programming for Boolean expression (TRUE or FALSE)



OPERATOR	DESCRIPTION
>	is greater than
<	is less than
>=	is greater than or equal to
<=	is less than or equal to
==	is equal to
!=	is not equal to

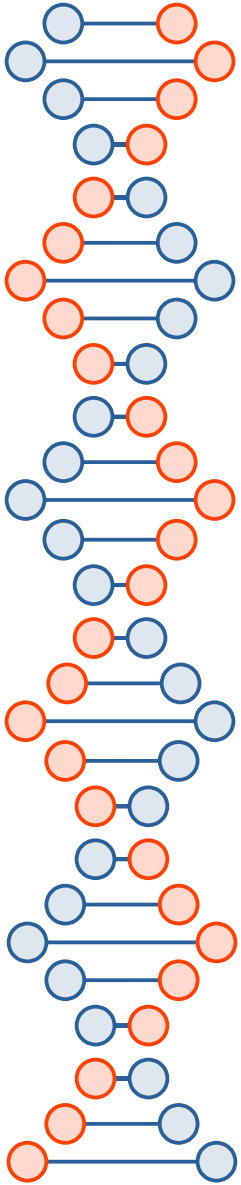
Flowchart Example: Comparison Operators





Compound Conditionals

- Conditionals that combine two or more Boolean expressions
- **Logical Operators** are used to combine the expressions
  - Most common are: AND, OR, and NOT

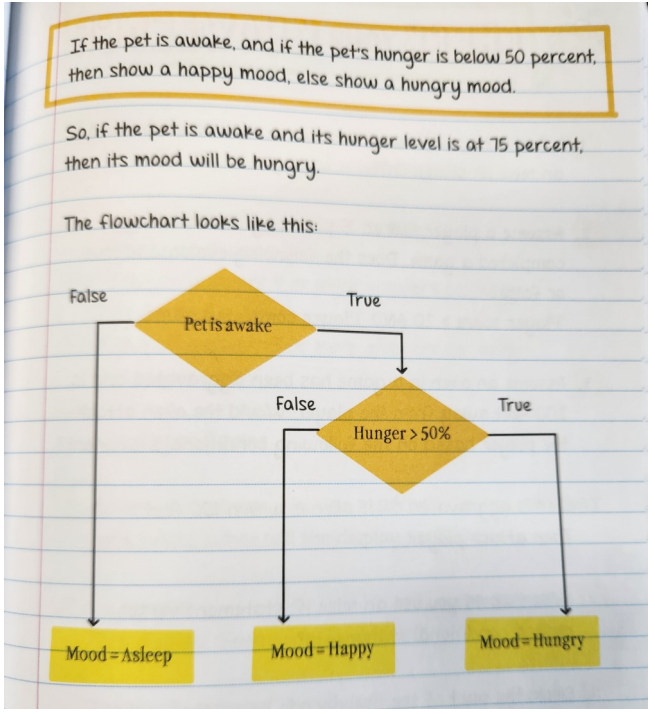


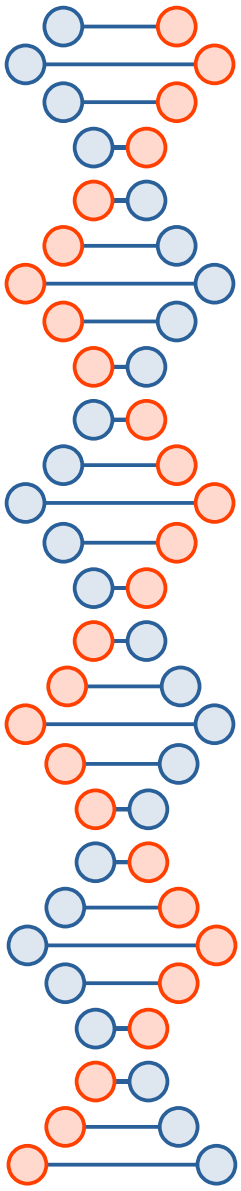
Logical Operators

- AND statements are true when both conditions are true
  - If one or both of the conditions is false, then the statement is false
- OR statements are tree if at least one of the conditions is true
- NOT statements don’t compare two conditions; instead it reverses the value from True to False and False to True

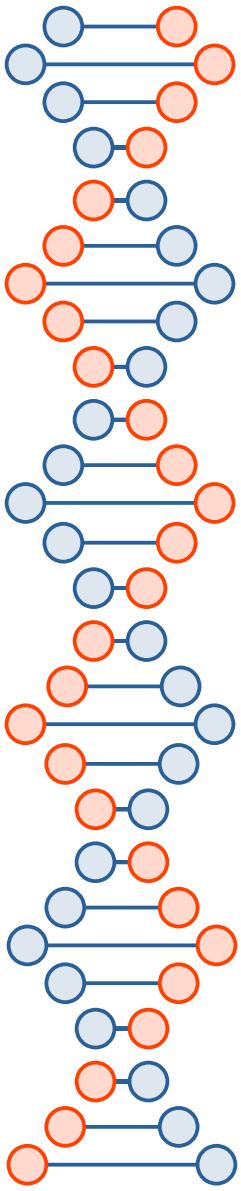
Nested Conditionals

- One conditional inside of another conditional





3) Loops

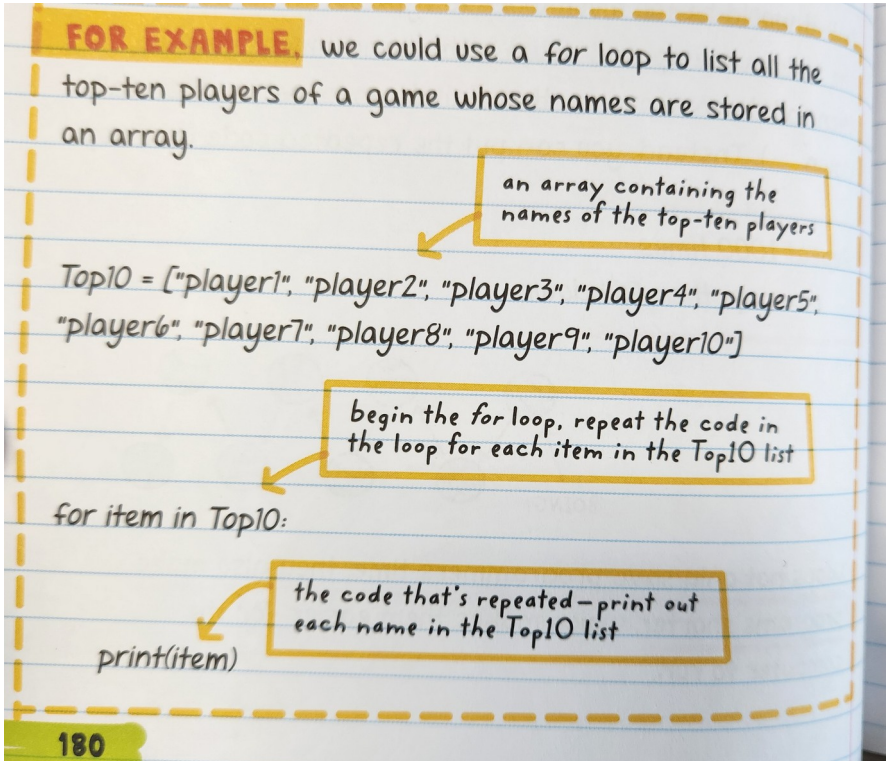


What is a Loop?

- Loop statements allow you to easily repeat a chunk of code many times
- Saves time and also makes programs shorter and simpler
- Different Types of Loops
  - FOR loop
  - WHILE loop
  - NESTED loop

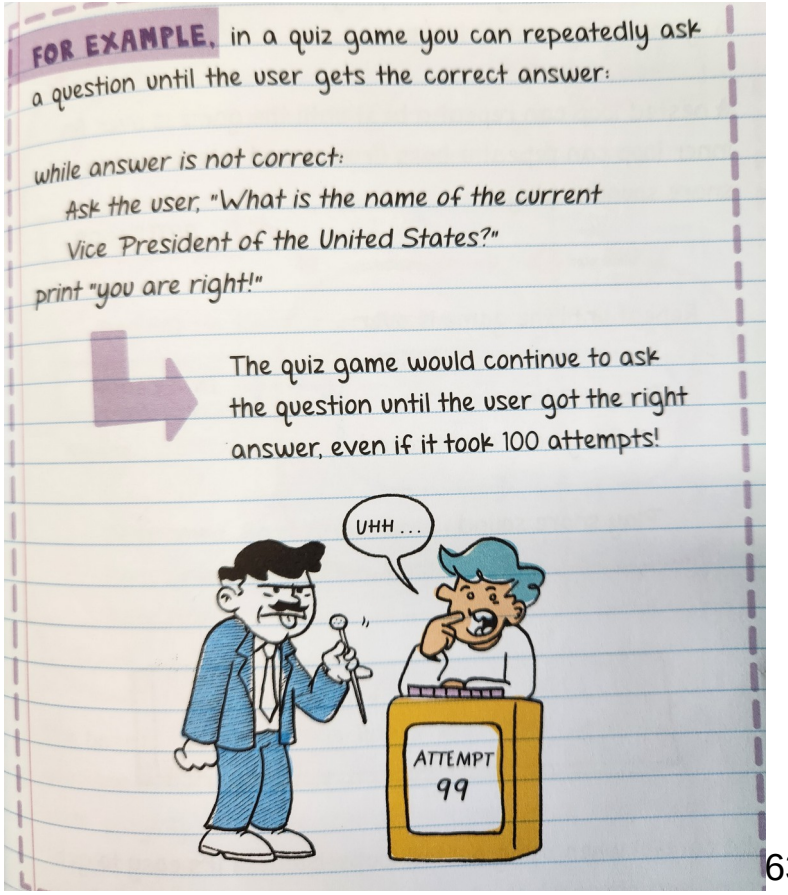
FOR Loop

- Repeats itself a set number of times
- Used when you know the number of times you want to repeat something
- FOR loops and arrays are often used together



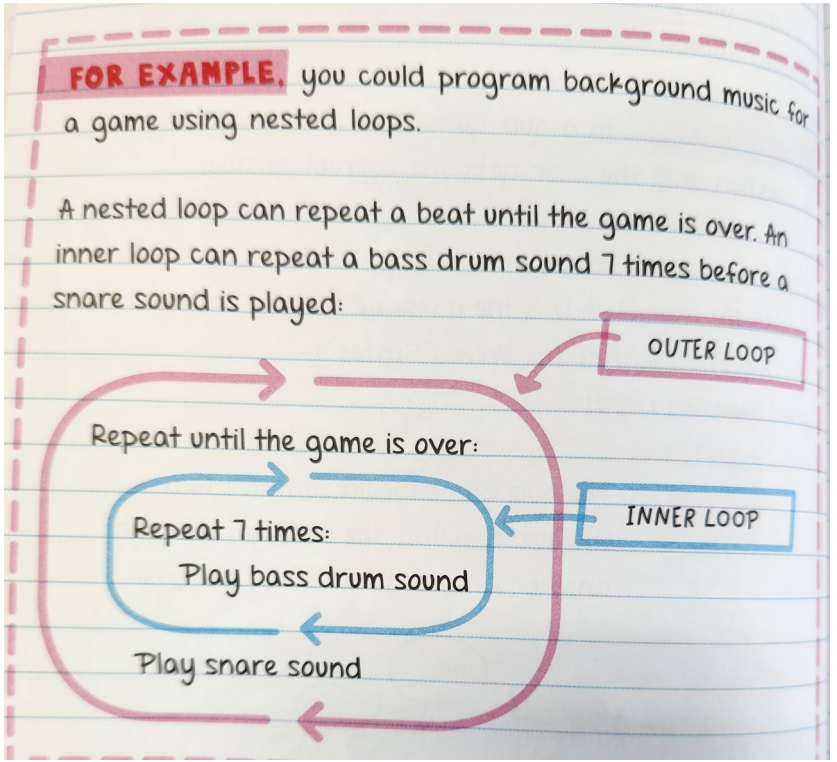
WHILE Loop

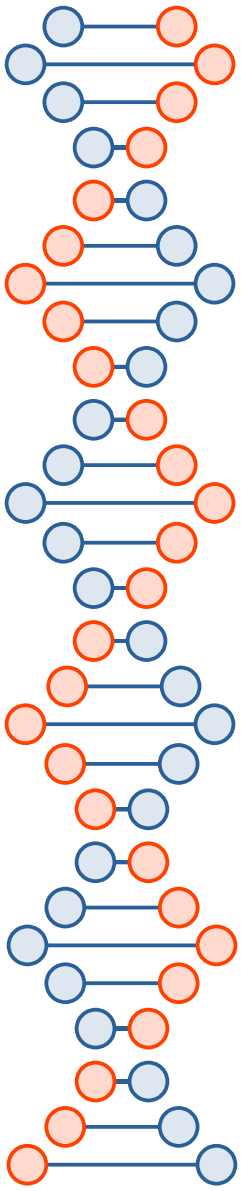
- Run a chunk of code until a condition is met
- Used when you don't know the exact number of times you want the loop to repeat



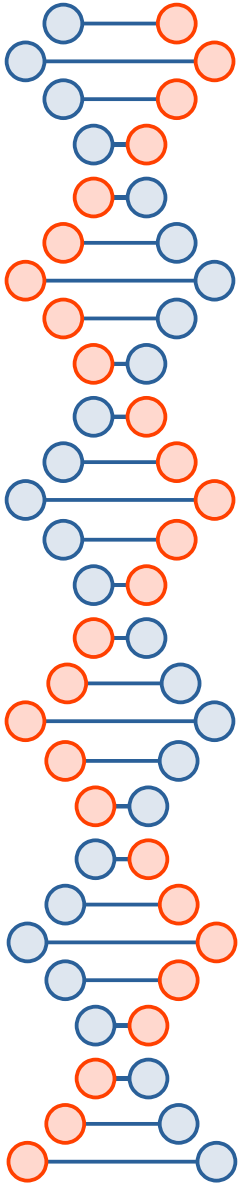
NESTED Loop

- When one loop is put inside another loop
- The inner loop always has to run through all its sequence before the outer loop code is run again



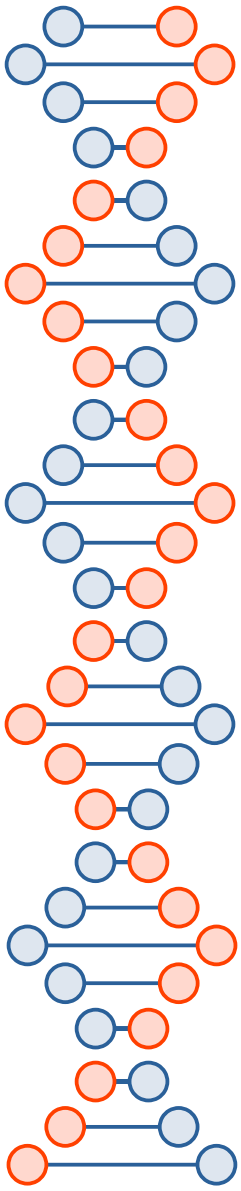


4) Events



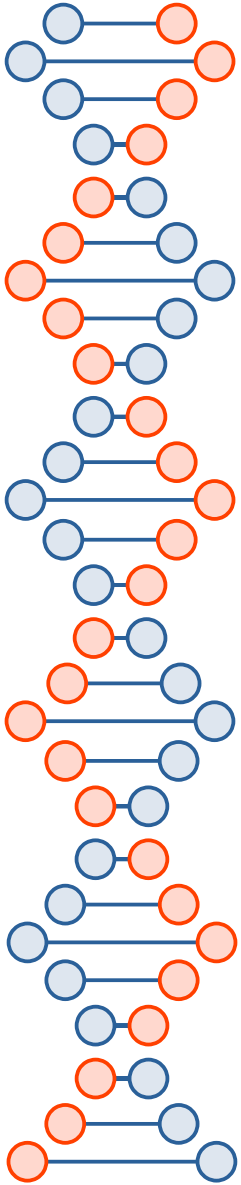
What is an Event?

- Actions that cause something to happen within a program
- Makes the programs interactive because the user is in control
- Example: clicking a mouse, pressing a key, tapping a touch screen

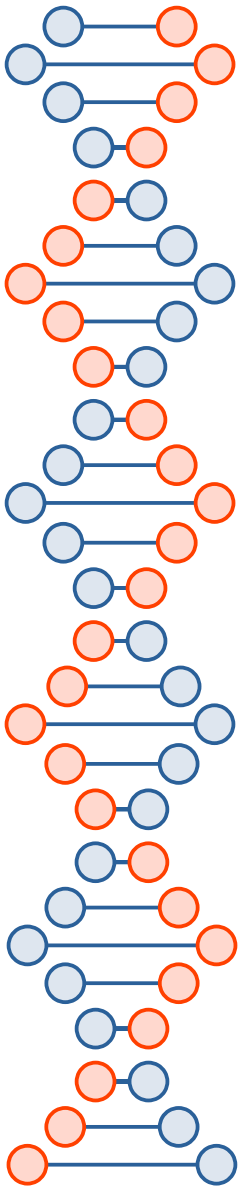


More on Events

- Can be external actions from a user
- Can be internal actions that happen within the program
- **Event Handler**: the code that is run when an event happens
  - Example: pressing the spacebar in a game is an event, the event handler is the code in the game that tells the character to jump when the spacebar is pressed



5) Procedures

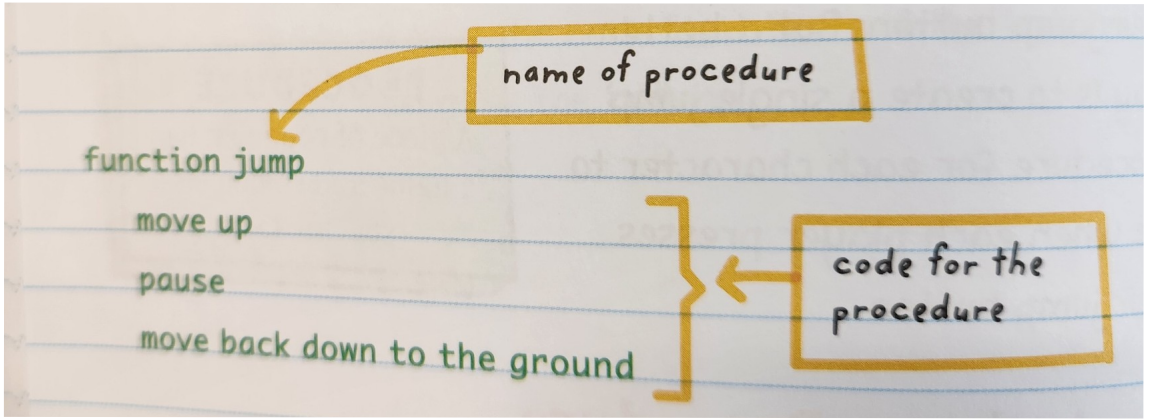


What is a Procedure?

- A piece of code that you can easily use over and over
- A piece of code that has a name and completes a specific task
- Alternative to loops
- Used when you want to use the same chunk of code in different parts of a program

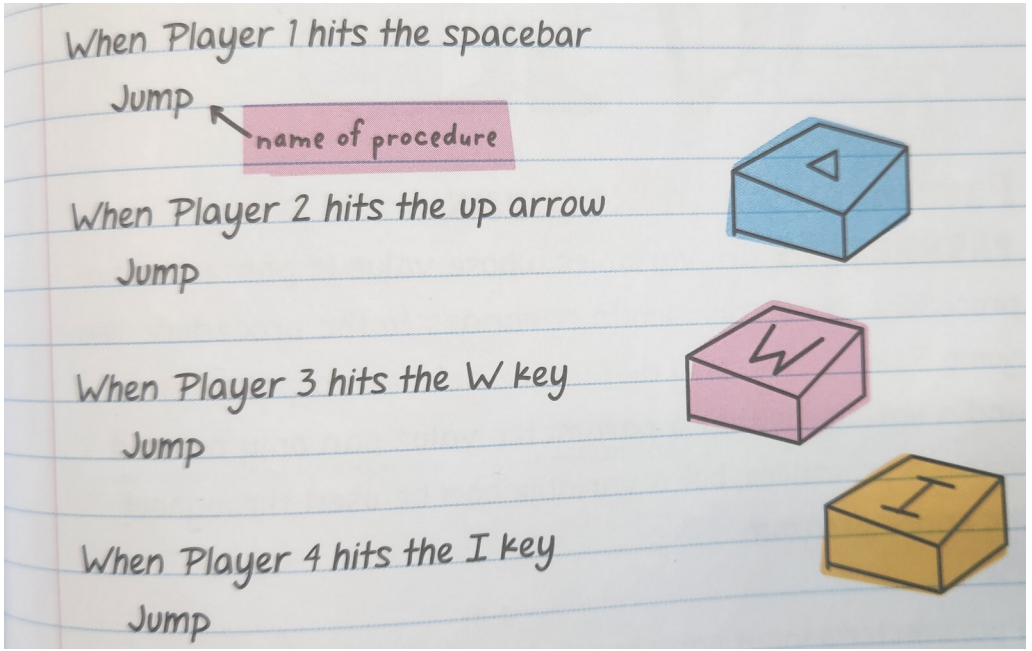
Declaring a Procedure

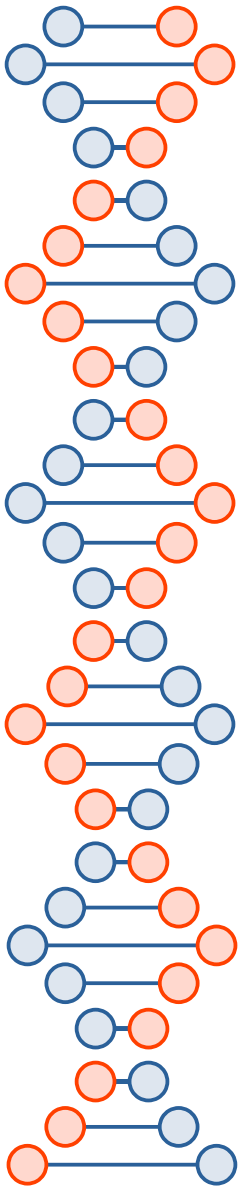
- Declare it by giving it a name and adding the code you want to use as a procedure
- Step 1: Identify the code to be saved
- Step2: Name the code and save



Calling a Procedure

- When you use the name of a procedure in a program so that it can run the code



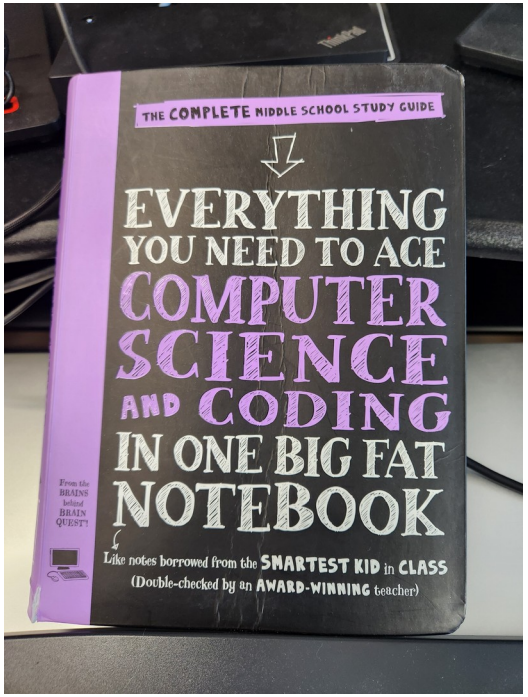


Parameters and Returns

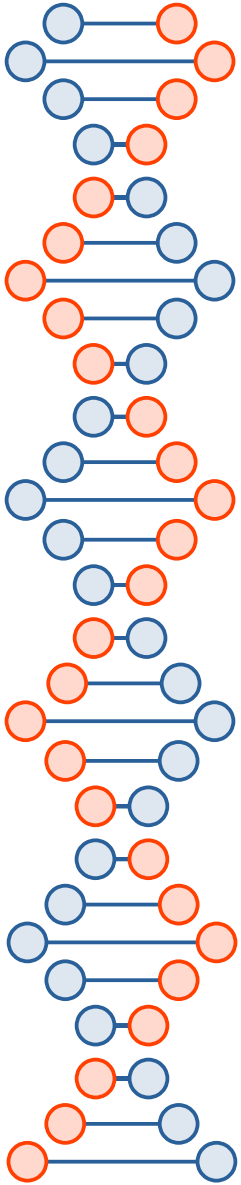
- Parameters: variables whose value is passed into a procedure
- Can only be used within the procedure

- Return value: the procedure output
- Can get a different value each time the procedure is run

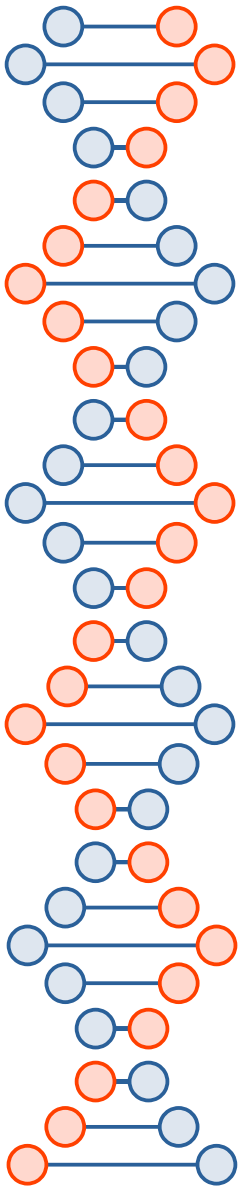
Reference



- For a more fun summary of this information, consider investing in this book
- Source of most of the images in the slides
- Also includes section of introductory python programming



Questions?



Next Workshop: March 31st

- You need to have Linux Subsystem for Windows on your computer. Contact Marc because you need administrator permissions to install it
- It will be hands-on so come ready to learn and try things out for yourself as we go
- Starting to learn how to use Linux CLI