CSC221 Project 2   
Design Document Ideas

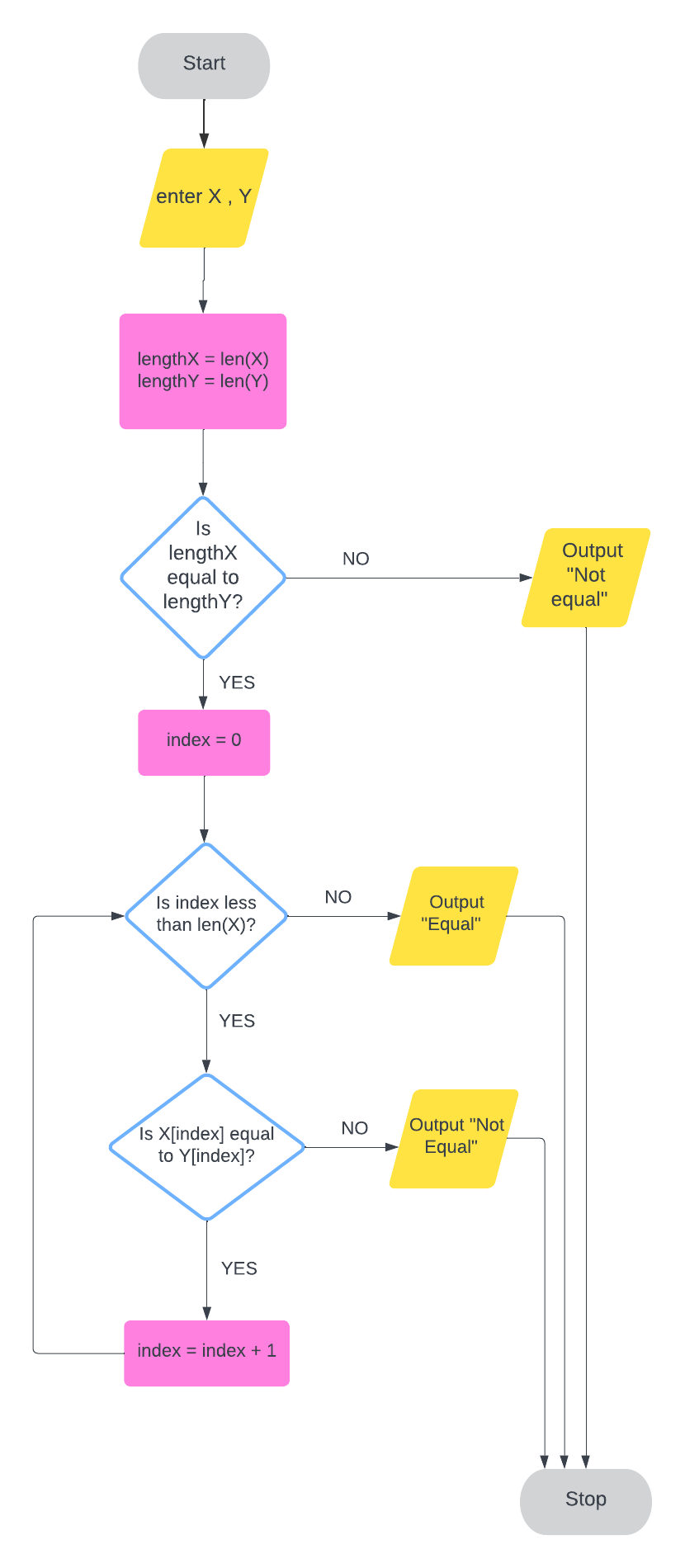
What your design document should do:

* Plan a solution for the project
* Express your solution as an algorithm using primitives like the ones discussed in Chapter 5 of your text

What you could include in your design document:

* Overview of your software
  + General functionality
  + What problem the software is addressing
* Analysis of your software
  + Information about problem/objective
  + Requirements for the software
    - “What does the solution look like?”
  + Information that pertains to potential solutions
* Algorithm to be used in the software
  + Expressed in pseudocode, flowcharts, diagrams
    - If using Object-Oriented programming, UML Diagrams
  + How you discovered the algorithm
    - Top-Down? Bottom-Up?

Expressing Algorithms

* Pseudocode
  + Mix of code and natural language
  + Example 1:   
    The algorithm expressed by the pseudocode below calculates a daily value from a yearly value by using division and conditional statements. Some syntax of Python is used (keywords, indentations, assignment operators), but natural language is used in variable names and the Boolean expression.
* Flowchart: diagram that visually express the flow/step of an algorithm
  + Charts that use symbols and arrows to express algorithms
  + Example: The algorithm described by the flowchart below compares two String inputs X and Y, checking whether they contain the same combination and order of letters.   
    You can see from the chart that the first step of the algorithm is to obtain X and Y, then get the length of each letter, and then to compare them. If X and Y have different lengths, then they cannot be equal. If X and Y have the same length, each letter (starting from the first one) is compared for equality. The use of conditional and repetition statements are identified by the flowchart.  
    
* UML Diagram: Diagram that shows the state and behavior of a class, and relationships between different classes
  + Example: There are two classes, Card and DeckOfCards.   
    A Card object contains two data values, numValue and Suit. It can also call the method getNum() on itself.   
    A DeckOfCards object contains a list called deck. deck is a list that contains 52 Card objects. A DeckOfCards object can call the methods shuffle() and deal() on itself. The deal() method returns a Card Object.  
    The Card class and the DeckOfCards class are related because the DeckOfCards class has a dependency on the Card class. This is illustrated by the line between the two classes.  
    