# Searching

# Searching for a number

**The problem…**

- You have been asked to design an algorithm that takes a list of numbers and a number being searched for. When the number is found in the list, the algorithm should return the index (or position) the number was at in the list.
- Recall the first element in the list is at index (or position) 0

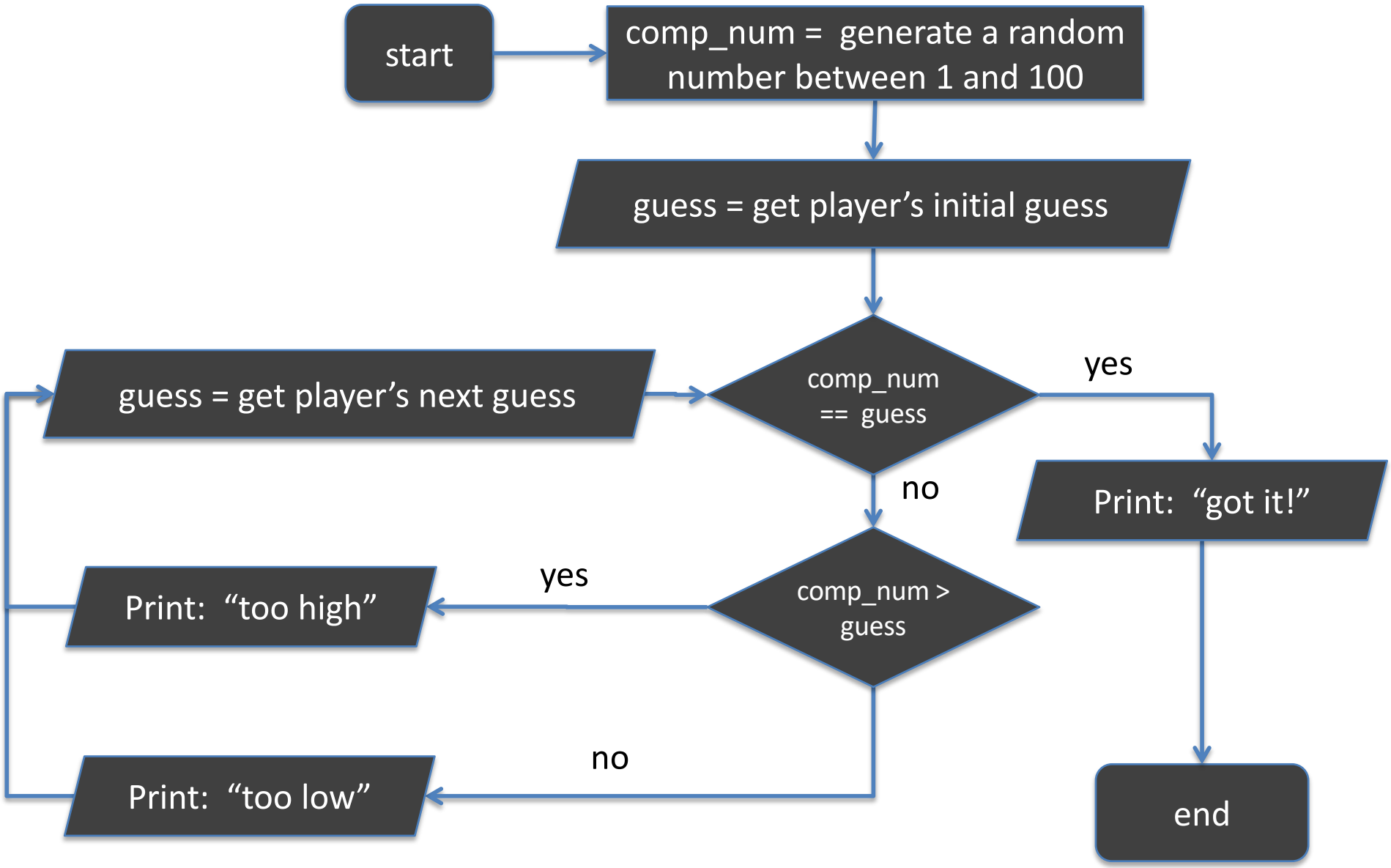
**Our process…**

- analyze the problem
  - identify the input(s)
  - identify the output(s)
  - articulate the purpose of the algorithm in terms of the input(s) and output(s)
  - articulate any assumptions
- work through example(s) of the problem
- design/implement an algorithm solution
- test your solution

# Let’s play a game

- A number between 1 and 100 is generated
- The player tries to guess the number.
- Each time the player guesses, the player is told if their guess is correct, too high, or too low.

# flowchart


start
comp_num = generate a random number between 1 and 100
guess = get player's initial guess
comp_num == guess
yes
Print: "got it!"
no
guess = get player's next guess
comp_num > guess
yes
Print: "too high"
no
Print: "too low"
end

# Algorithm exercise: guessing game

**Assumptions**: **comp_num** will be an integer taken at random between 1 and 100
**players_guess** will be an integer between 1 and 100.

comp_num = an int selected at random between 1 and 100
players_guess = an int entered by player between 1 and 100

**WHILE** players_guess != comp_num

Check **IF** players_guess > comp_num
**PRINT** "Your number is too high. Guess again."
**ELSE**
**PRINT** "Your number is too low. Guess again."

players_guess = an int entered by player between 1 and 100
// player enters a new integer in players_guess and it is
// checked for equality with comp_num
**END WHILE**
**Print** "You got it – finally!"