Unit 1: Problem Solving

Part 4: Subprocesses

Topic 2: Creating Algorithms

Lecture Contents

- Quick Review of Algorithms and Flowcharts
- Flowchart Symbol: ***Subprocess***
- Pseudocode: ***Procedures*** and ***Functions***
- Reading:
  - review pages 3 to 14
  - Appendix 2: Flowchart Symbols, page 293
  - Appendix 3: SUBPROGRAMS, page 297

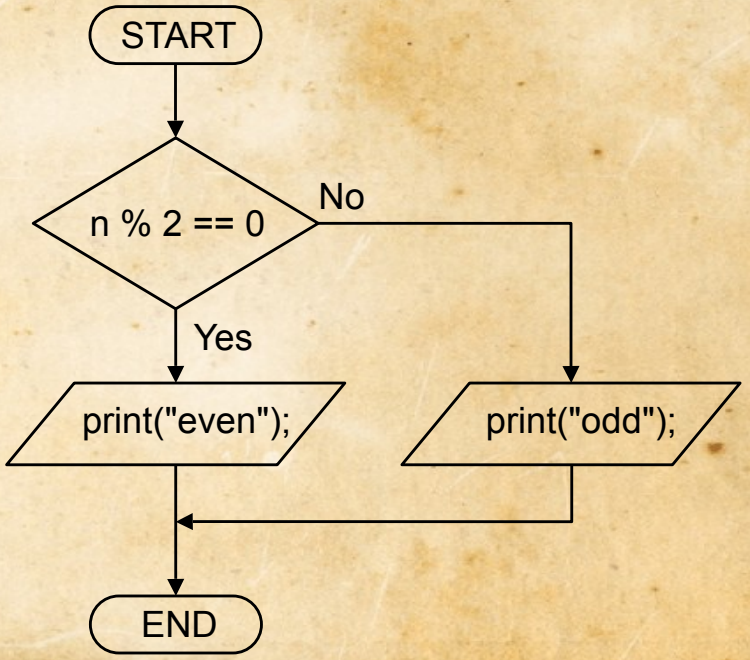
Algorithm - Definition

- Algorithm: an ***unambiguous sequence*** of steps to solve a problem or perform a task.
  - ***Unambiguous*** : clear and precise with no room for misinterpretation
  - ***Sequence*** : and *ordered* set
- The result of following a *successful* algorithm should always be the same (***consistency***)

  (given the same input)

Flowcharts

- Flowcharts are a diagrammatic way to show an algorithm.
- The visual aspect may aid in understanding



Flowchart Symbols

- You should be familiar with the following symbols:

- **Flowlines**
  - Maximum one output from any block

- **Terminal** (start/end)
  - Start only output, end only input
  - Diagram might not have an end

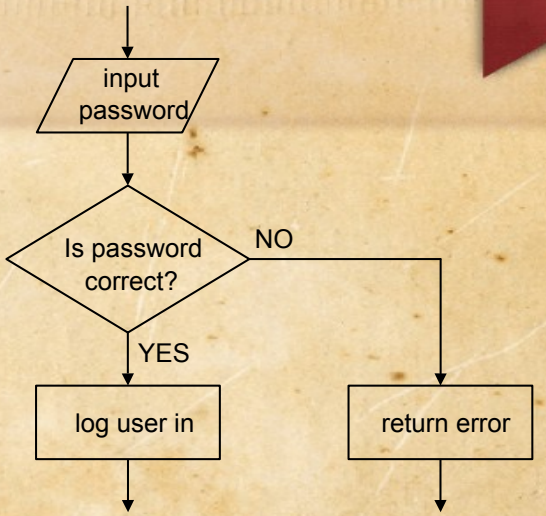
- **Processes**
  - Action (verb)

- **Input / Output**
  - Ensure it does not look like a rectangle!

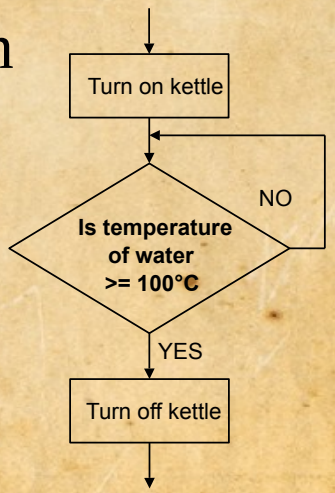
- **Decision**
  - Always <u>two</u> outputs

Flowchart Symbol – ***Decision***

- A ***selection*** has two two alternative paths



- A *selection* that makes a loop is called an ***iteration***.
  - As a verb, we ***iterate***.



Flowchart Symbol – ***Subprocess***

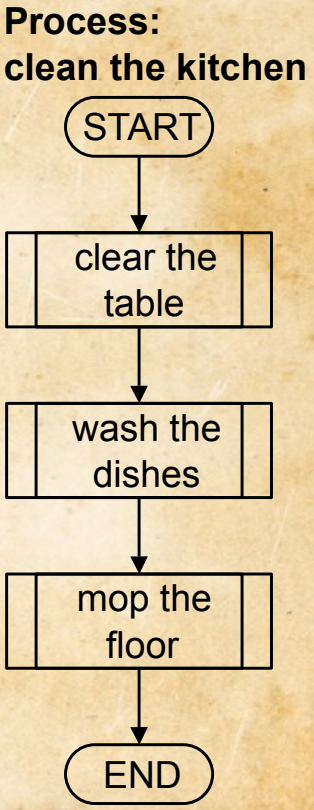
- Predefined Process
  - A ***subprocess*** (also: ***subroutine***, or in programming a ***function*** or ***method***)
  - The details (steps) of the *subprocess* are shown in a different flowchart
    - the hiding of details is called ***abstraction***.

Flowchart Symbol – ***Subprocess***

- For example… if we write an algorithm to clean the kitchen…

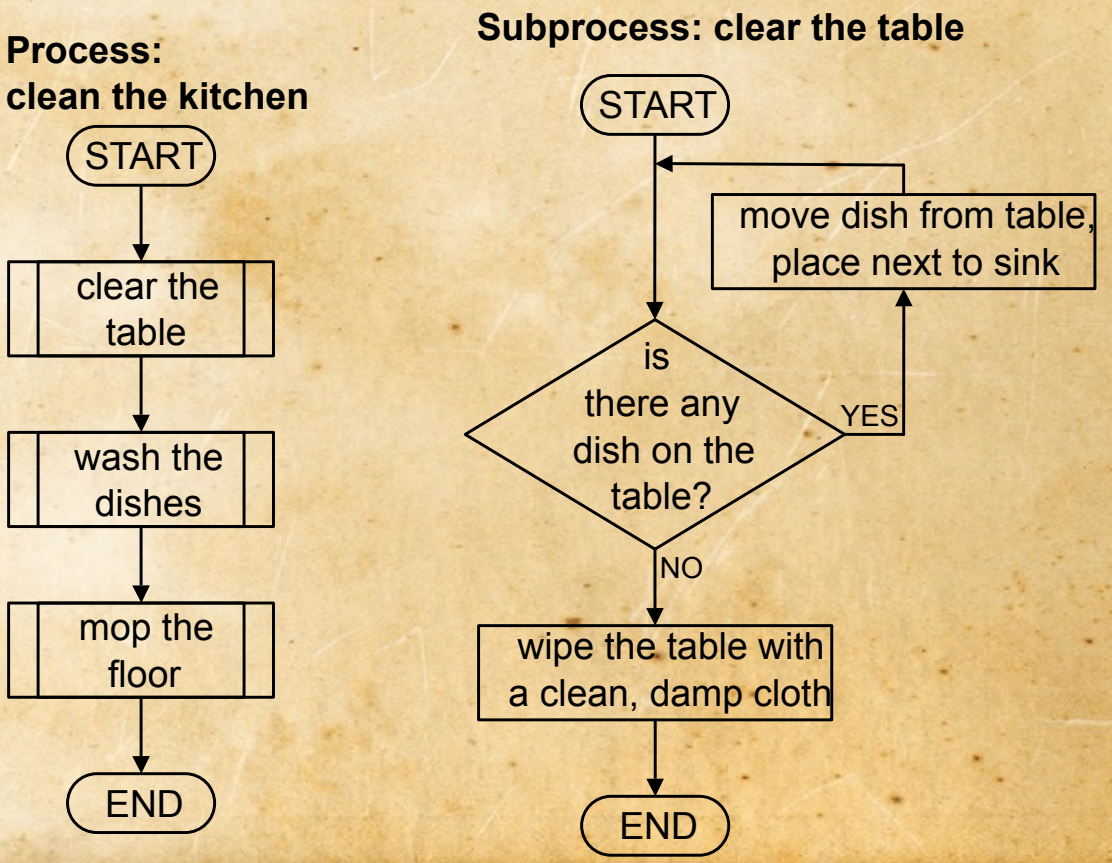
Flowchart Symbol – ***Subprocess***

- For example… if we write an algorithm to clean the kitchen…
  - To start, we might ***abstract*** (hide the details of) the steps in cleaning the kitchen



Flowchart Symbol – ***Subprocess***

- For example… if we write an algorithm to clean the kitchen…
  - To start, we might ***abstract*** (hide the details of) the steps in cleaning the kitchen
  - Then we can ***decompose*** (break down) those steps in a separate flowchart



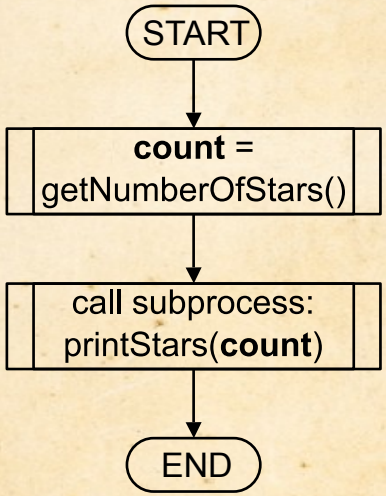
Flowchart Symbol – ***Subprocess***

- A program that prompts the user to enter a number of stars, then prints the number of stars the user requests.

Flowchart Symbol – ***Subprocess***

- A program that prompts the user to enter a number of stars, then prints the number of stars the user requests.

Main Process:

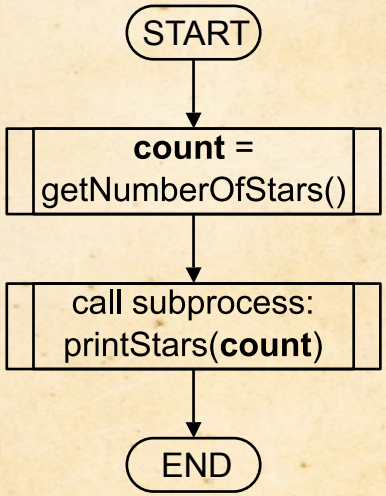


Subprocess: getNumberOfStars

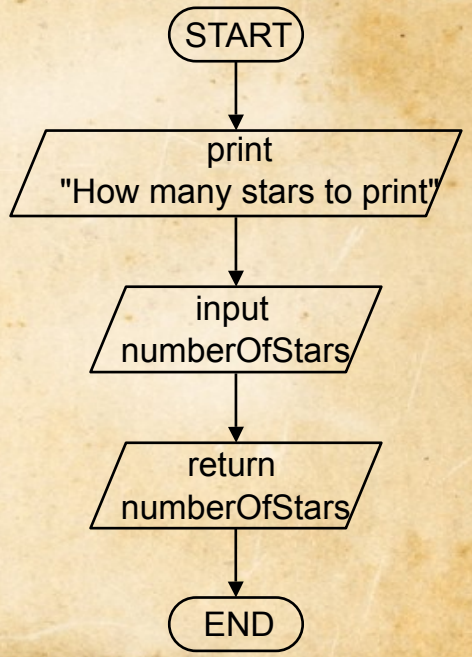
Flowchart Symbol – ***Subprocess***

- A program that prompts the user to enter a number of stars, then prints the number of stars the user requests.

Main Process:



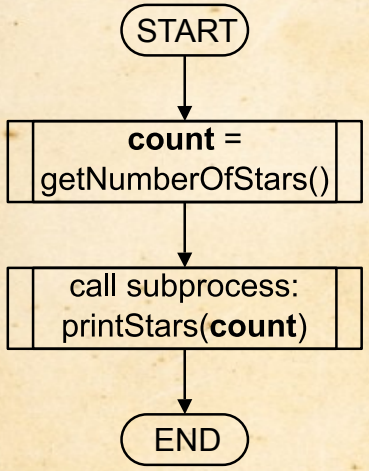
Subprocess: getNumberOfStars



Flowchart Symbol – ***Subprocess***

- A program that prompts the user to enter a number of stars, then prints the number of stars the user requests.

Main Process:

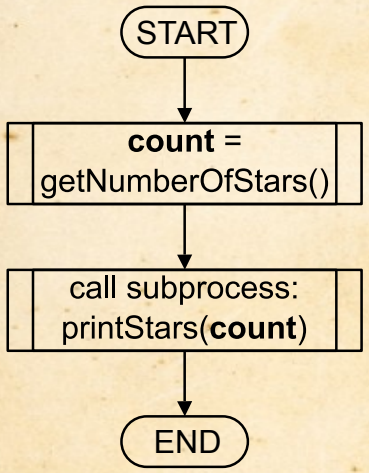


Subprocess: printStars

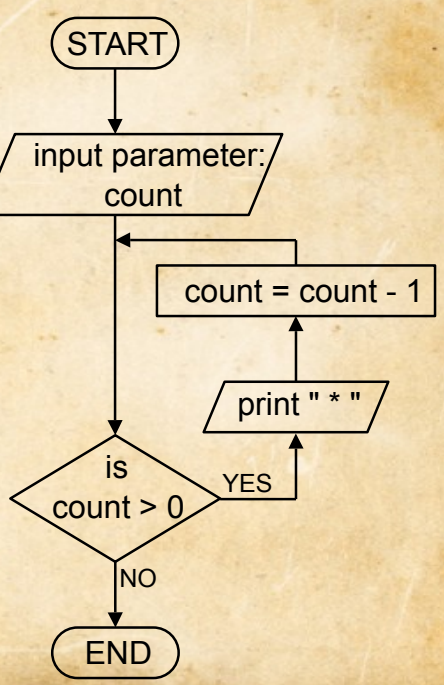
Flowchart Symbol – ***Subprocess***

- A program that prompts the user to enter a number of stars, then prints the number of stars the user requests.

Main Process:

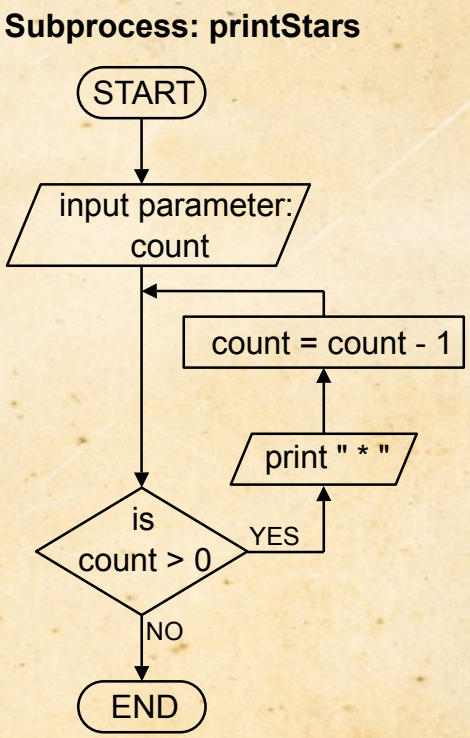


Subprocess: printStars



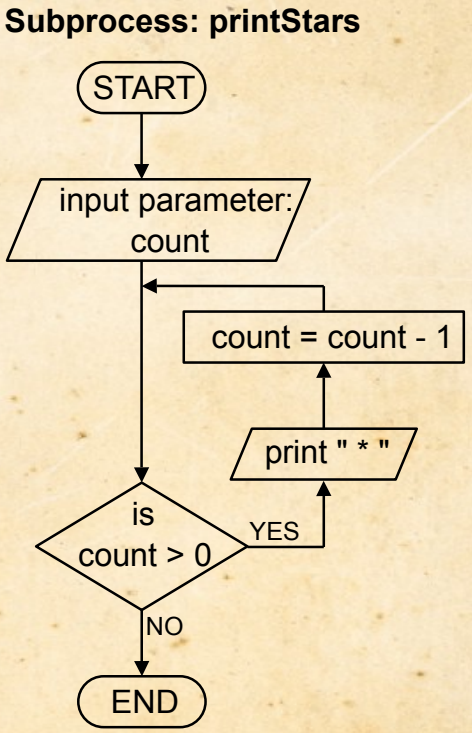
Pseudocode – Procedure

- A ***procedure*** can take input ***parameters***, but does <u>not</u> return a value



Pseudocode – Procedure

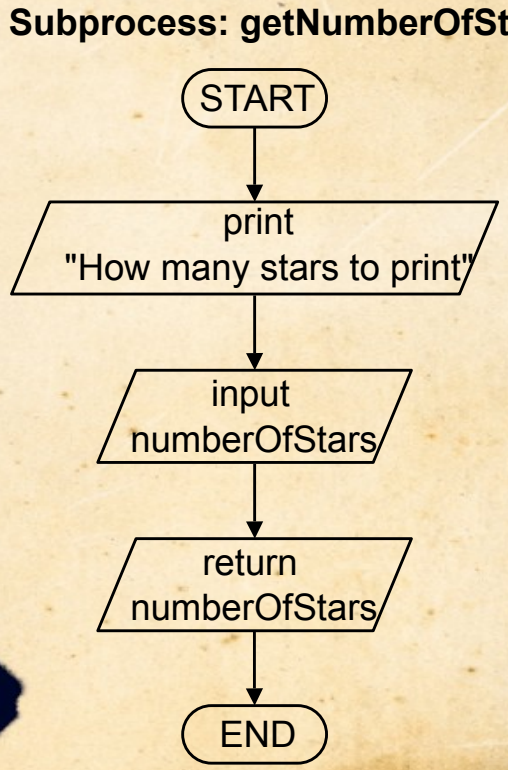
- A ***procedure*** can take input ***parameters***, but does <u>not</u> return a value




```
PROCEDURE printStars ( count )
BEGIN PROCEDURE
    WHILE (count > 0) DO
        SEND “*” TO DISPLAY
        SET count TO count – 1
    END WHILE
END PROCEDURE
```

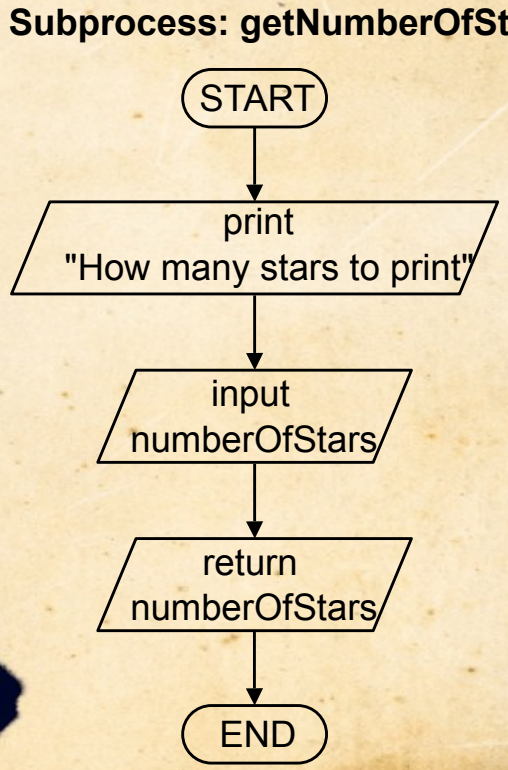
Pseudocode – Procedure

- A ***function*** can take input ***parameters***, and returns a value



Pseudocode – Procedure

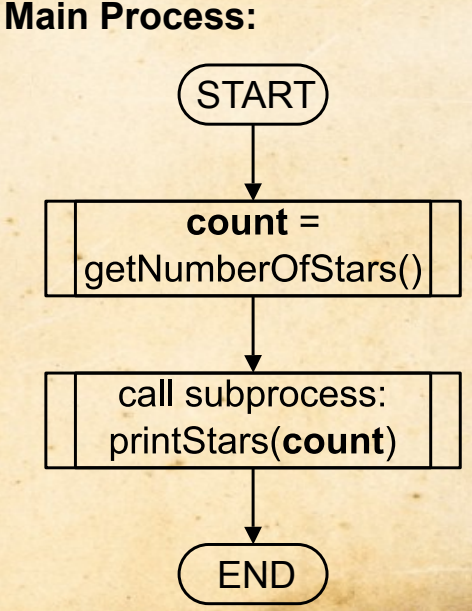
- A ***function*** can take input ***parameters***, and returns a value




```
FUNCTION getNumberOfStars( )
BEGIN FUNCTION
    SEND “How many stars to print” TO DISPLAY
    RECEIVE numberOfStars FROM KEYBOARD
    RETURN numberOfStars
END FUNCTION
```

Pseudocode – Calling Functions and Procedures

- Here is the pseudocode for the main flowchart:
  - When we want to run the code in a *function* or *procedure*, we say we ***call*** the *function*, or ***call*** the *procedure*




```
SET count TO getNumberOfStars()

printStars( count )
```

Unit 1: Problem Solving

Part 4: Subprocesses

Topic 2: Creating Algorithms