Design: Standard Algorithms

Standard Algorithms

There are three standard algorithms you need to know for National 5:

- Running Total
- Input Validation
- Traversing a 1D Array

Running Total

The Running Total standard algorithm uses a fixed or conditional loop, taking in values from a user and adding them together.

There are 4 steps to a running total algorithm:

```
1. Set total variable to 0
2. Start loop
3.     Ask the user to enter a number
4.     Add the number to the total
```

Running Total: Pseudocode

With a fixed loop

```
SET total TO 0
FOR loop = 1 to 20
    RECEIVE amountRaised FROM KEYBOARD
    total = total + amountRaised
END LOOP
SEND total TO DISPLAY
```

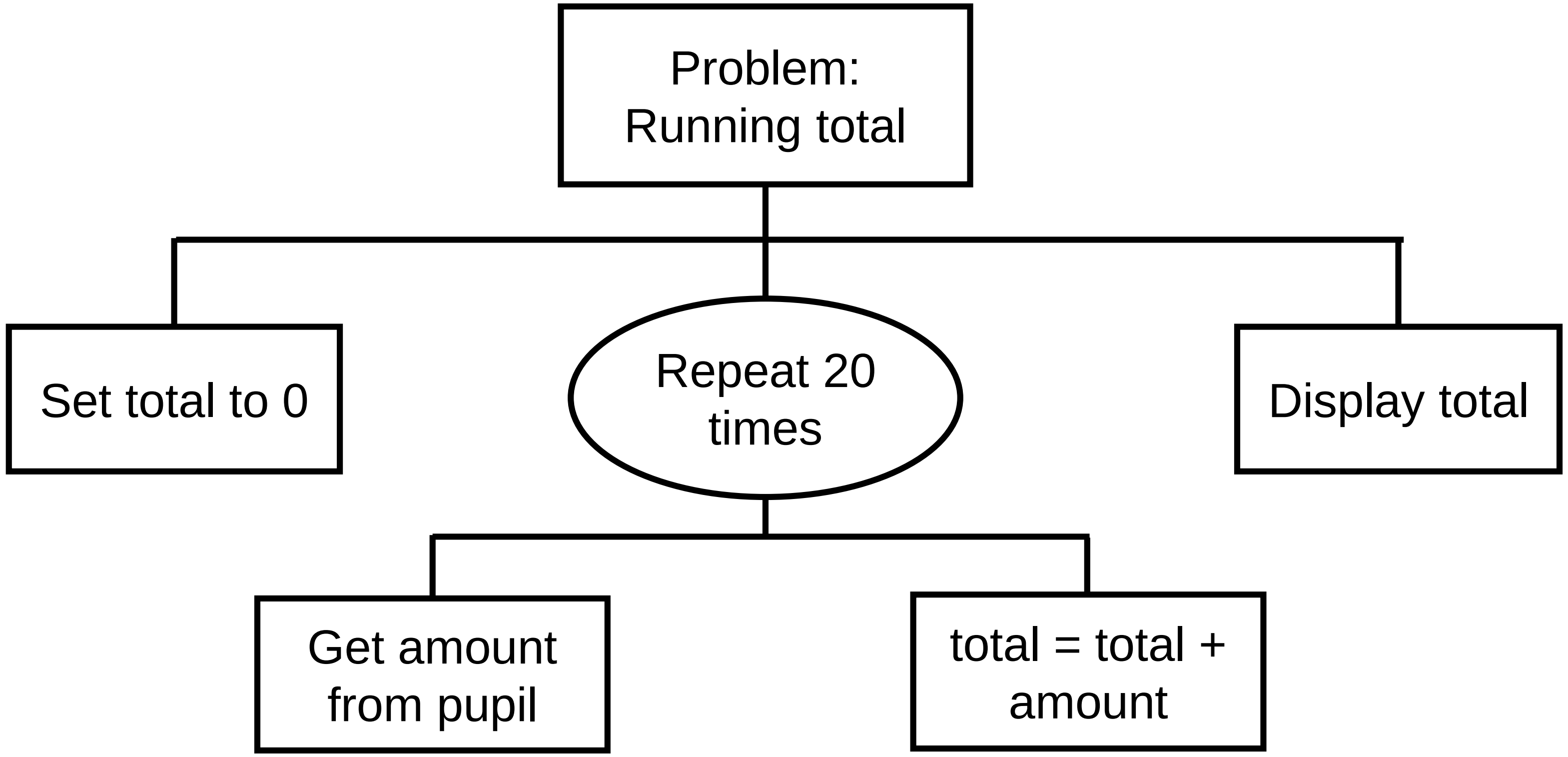
Running Total: Pseudocode

With a conditional loop

```
SET total TO 0
SET morePupils TO True
WHILE morePupils = True
    RECEIVE amountRaised FROM KEYBOARD
    total = total + amountRaised
    RECEIVE morePupils FROM KEYBOARD
END LOOP
SEND total TO DISPLAY
```

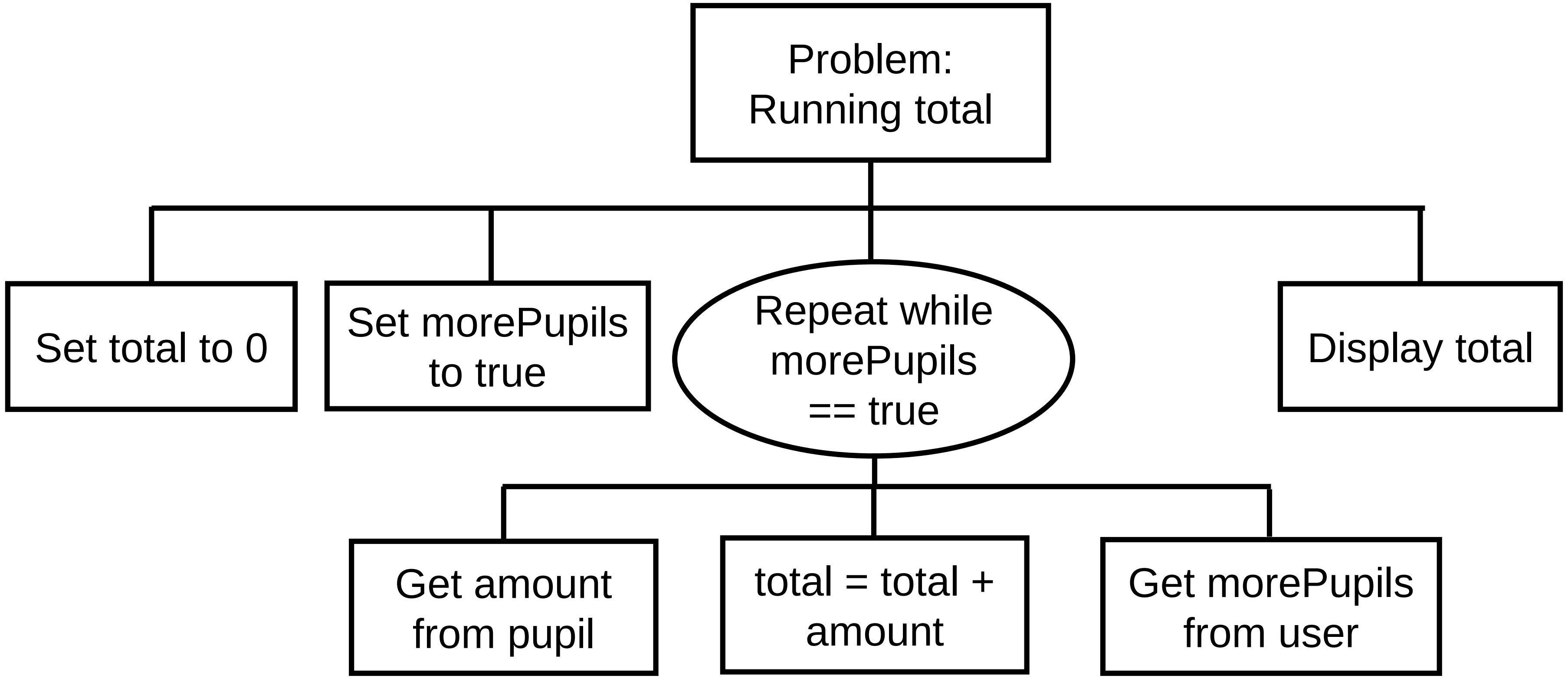
Running Total: Structure Diagram

With a fixed loop



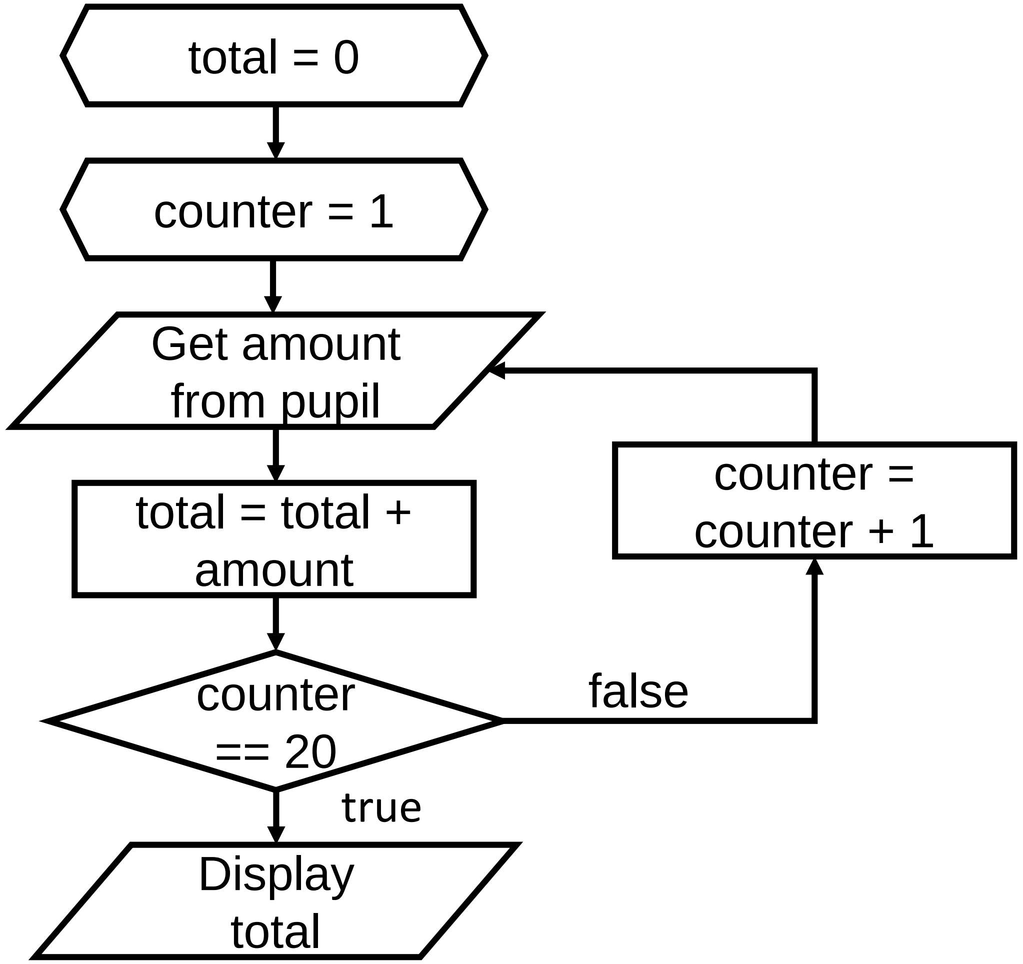
Running Total: Structure Diagram

With a conditional loop



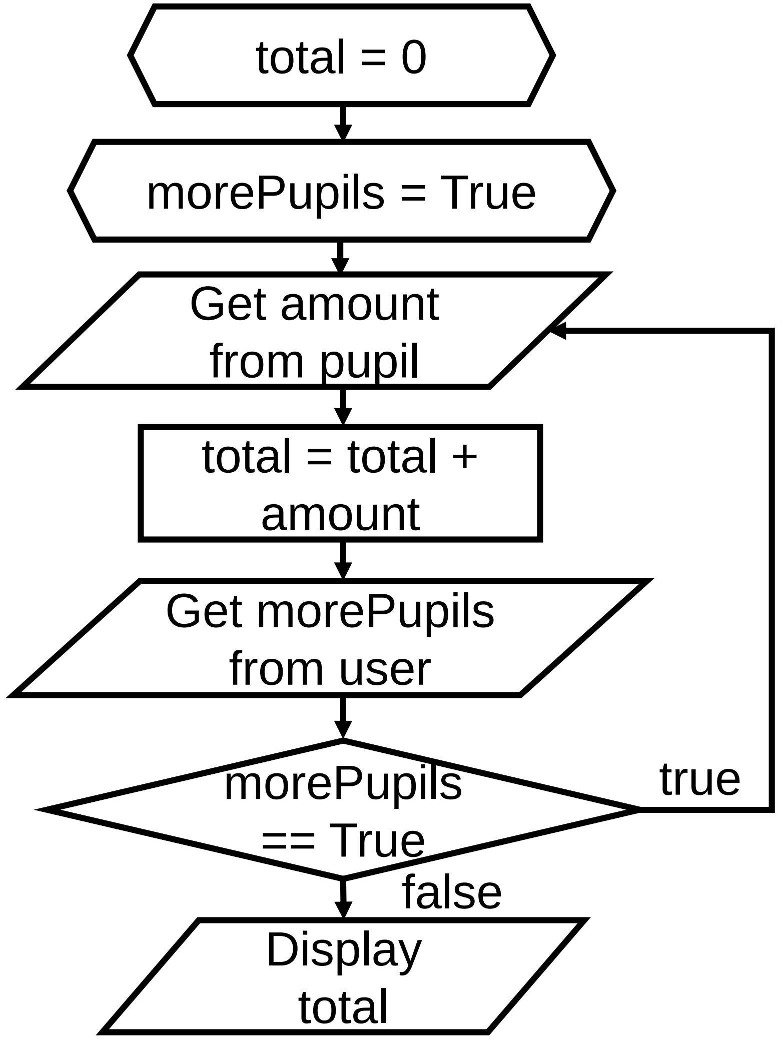
Running Total: Flowchart

With a fixed loop



Running Total: Flowchart

With a conditional loop



Running Total: Python

With a fixed loop

```
total = 0

for counter in range (0, 20):
   amount = int(input(“Enter amount raised”))
   total = total + amount

print(total)
```

Running Total: Python

<u>With a conditional loop</u>

```
total = 0
morePupils = “Y”

while morePupils == “Y”:
   amount = int(input(“Enter amount raised”))
   total = total + amount
   morePupils = input(“Are there more pupils? Y/N”)

print(total)
```

Input Validation

The Input Validation standard algorithm uses a conditional loop, and continues to ask the user for input until an acceptable answer is provided.

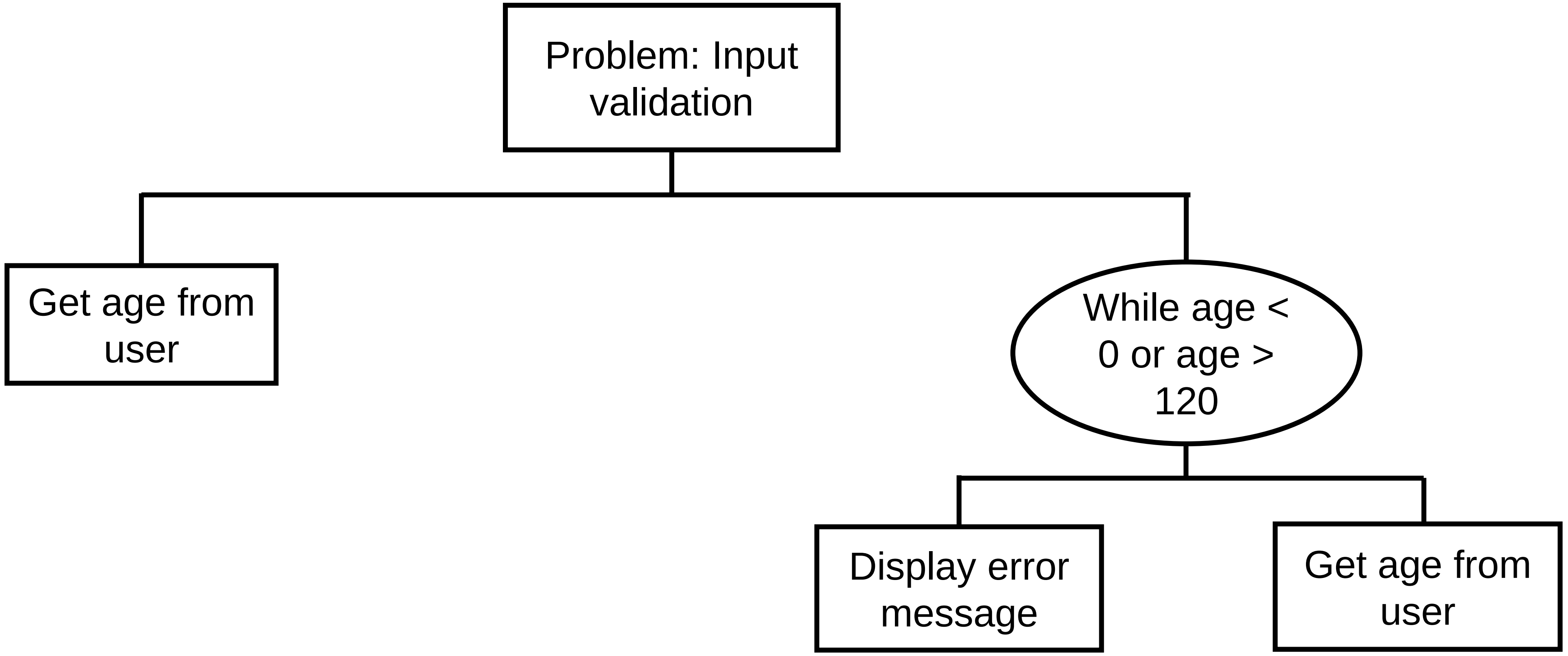
There are 4 steps to an input validation algorithm:

```
1. Receive value from user
2. Start conditional loop if value is unacceptable
3.     Display an error message
4.     Receive value from user
```

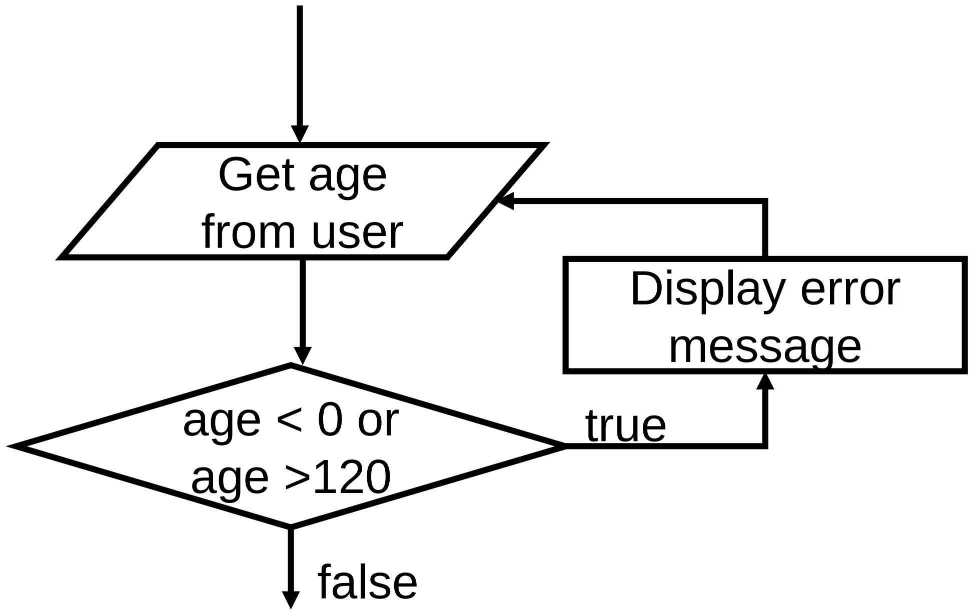
Input Validation: Pseudocode

```
RECEIVE age FROM KEYBOARD
WHILE age < 0 OR age > 120
    SEND “Invalid age.  Enter age between 0 and 99” TO DISPLAY
    RECEIVE age FROM KEYBOARD
END LOOP
```

Input Validation: Structure Diagram



Input Validation: Flowchart



Input Validation: Python

```
age = int(input(“Enter your age: ”))

while age < 0 or age > 120:
  print(“Invalid age – enter 0-99 only”)
  age = int(input(“Enter your age: ”))
```

Traversing a 1D Array

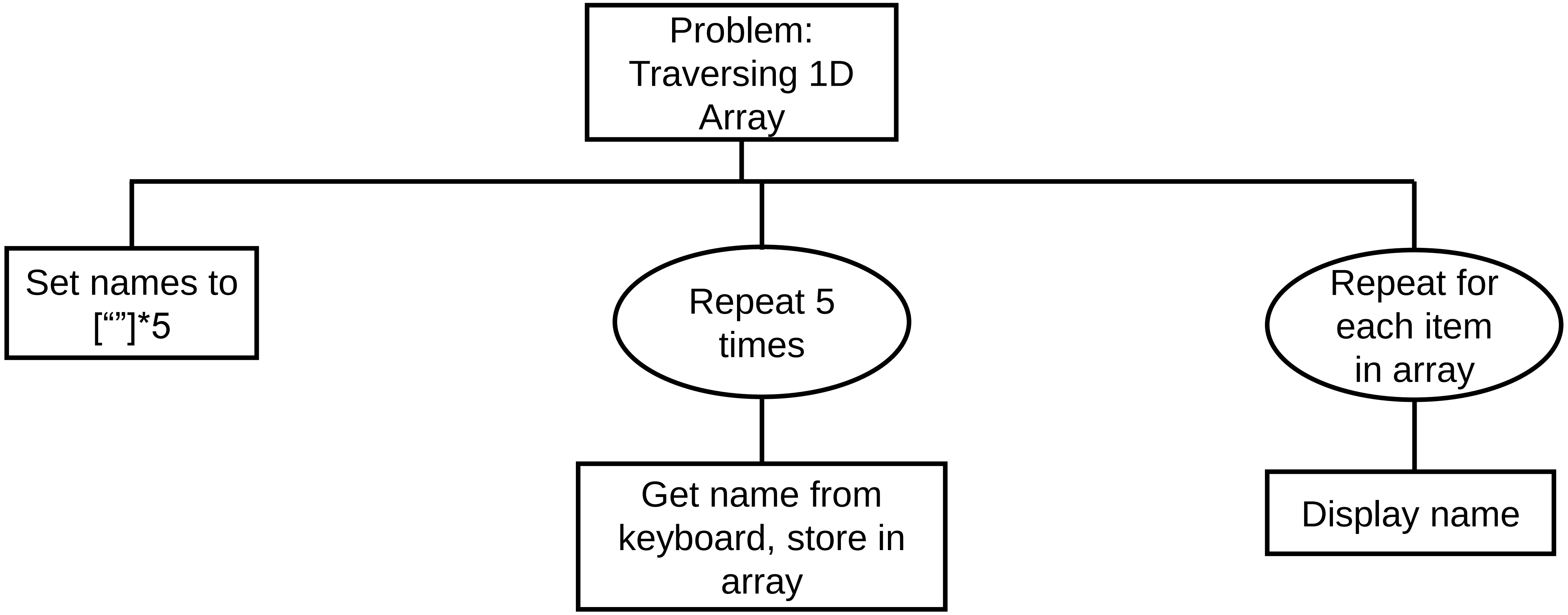
We can use the index of an array to move through the array within a fixed loop. This is called **traversing** the array.

This standard algorithm is usually used to add values to an array, or to display every item in an array.

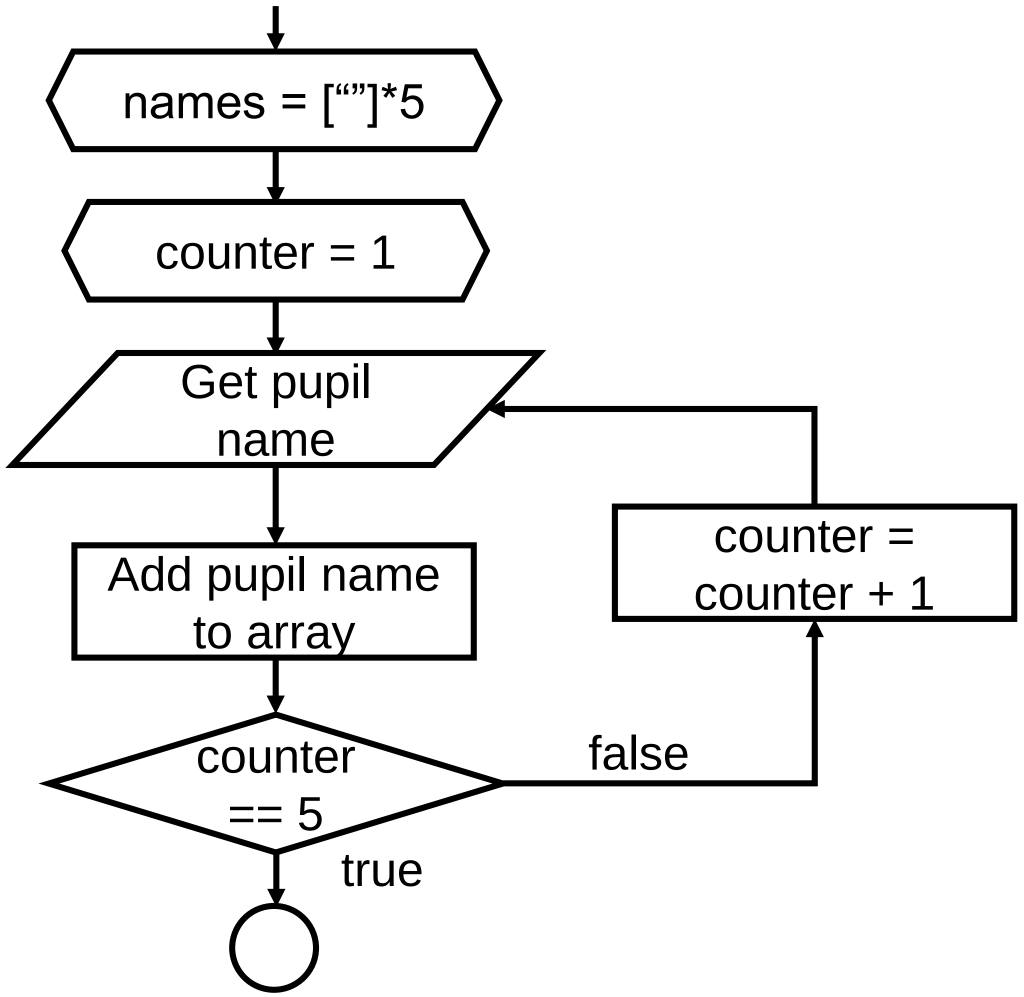
Traversing a 1D Array: Pseudocode

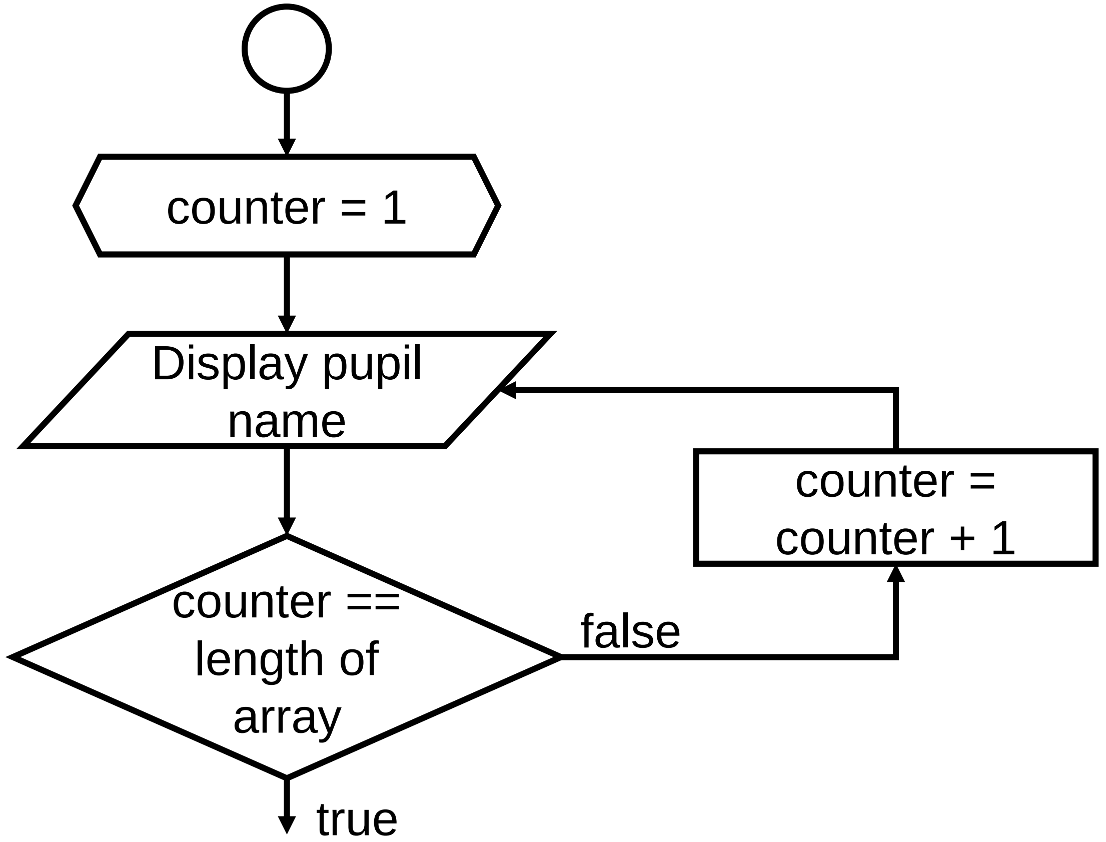
```
SET names TO array of strings
FOR loop = 1 to 5
    RECEIVE name FROM KEYBOARD
    STORE name in array
END LOOP
FOR each item in the array
    SEND name TO DISPLAY
END LOOP
```

Traversing a 1D Array: Structure Diagram


Problem: Traversing 1D Array
Set names to [“”]*5
Repeat 5 times
Get name from keyboard, store in array
Repeat for each item in array
Display name

Traversing a 1D Array: Flowchart





Traversing a 1D Array: Python

```
names = [“”]*5

for counter in range(0, 5):
   names[counter] = input(“Enter the pupil’s name.”)

for counter in range(0, len(names)):
   print(“Pupil name: ” + names[counter])
```

Traversing a 1D Array: Worked Example

```
#Traversing a 1D Array Example

#initialising variables
maxScore = 0

#initialising arrays and adding 20 indices
names = [""]*20
scores = [""]*20

#inputs - getting scores for each pupil in the class
for counter in range(0,20):
    names[counter] = input("Enter the pupil's name ")
    scores[counter] = int(input("Enter the pupil's score "))

#input - getting maximum score possible
maxScore = int(input("How many marks were available in the test? "))

#traversing an array using a fixed loop
for person in range(0,20):
    print(names[person] + " has scored " + str((scores[person]/maxScore)*100) + "%")
```