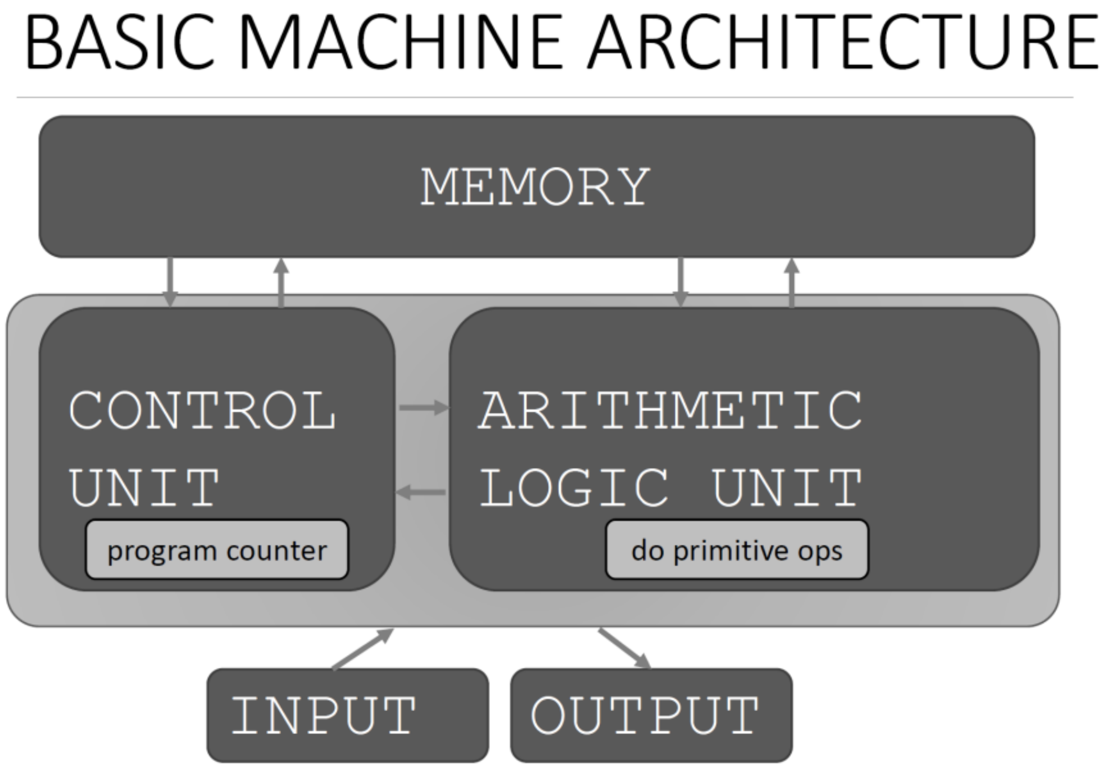
Program Design

Lecture 3

When We Design Algorithms Who Are We Talking to?



Varaibles and Assignment

- Algebra – Variable expressions - A variable is a placeholder that stands for an unknown quantity. E.g. a circle of radius r has the area:
  - circleArea = πr^2
  - In this expression r stands for any positive number. In order to calculate the area of a circle of radius 4 we simply set r = 4.
- In such cases expressions that contain variables are rules that describe how to compute a number when we are given values for the variables.
- In computing it is similar. We create rules that tell the computer how to produce some new data from some existing data.

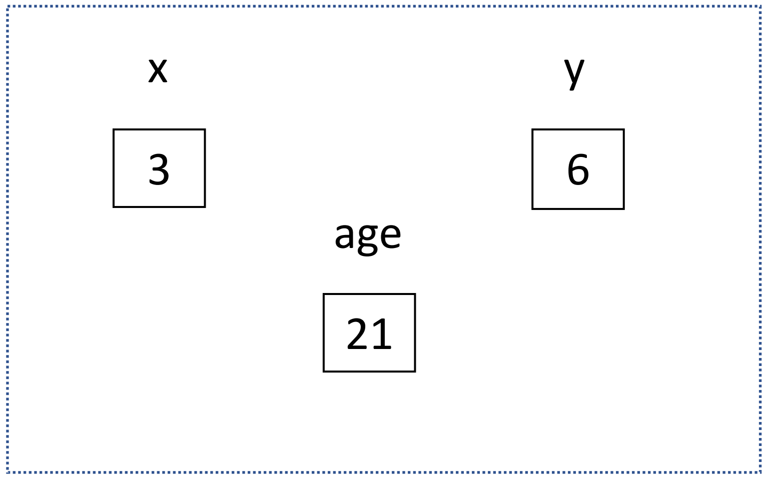
Assignements – Variables and Binding

- And assignment is the ‘binding’ of a value to a particular variable.
- An assignment looks like this
  - x=3
  - y=6
  - age = 21

  The variable on the left always receives the value on the right

Assignments:

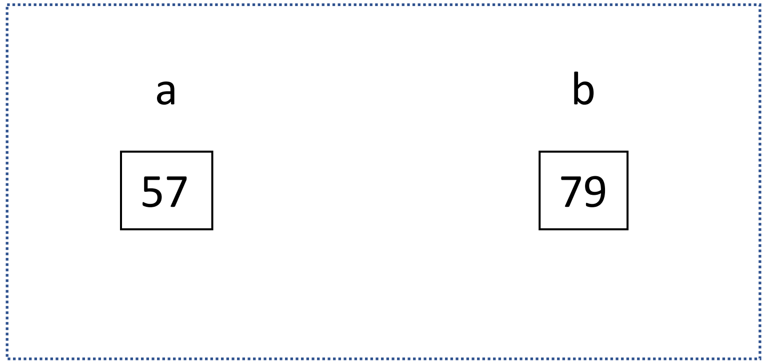
```
x=3
y=6
age = 21
```



Computer memory

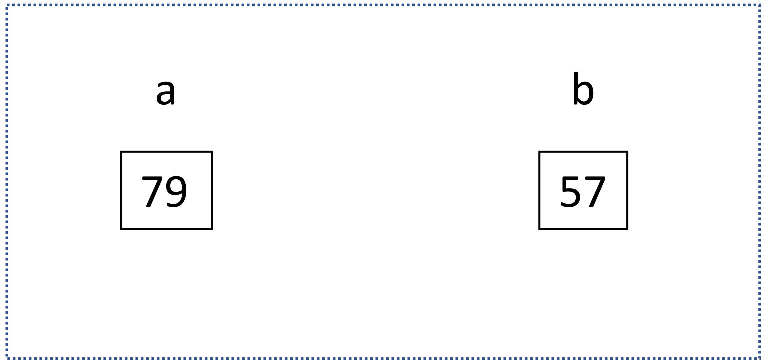
Exchanging 2 variables

Assignments:
a = 57
b=79



Computer memory

Target Configuration



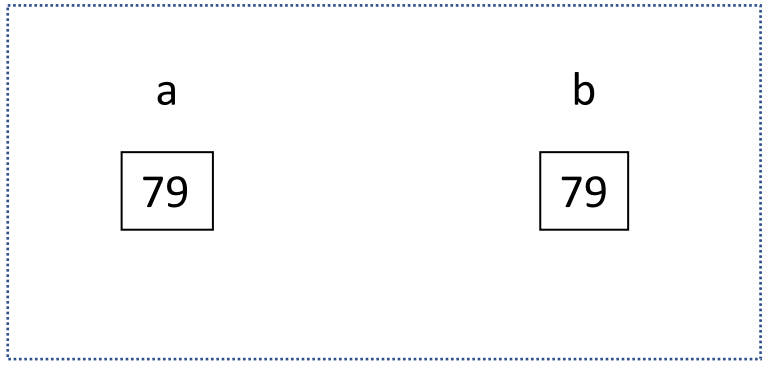
Computer memory

Algorithm:

```
a=b
b=a
```

Algorithm:

```
a=b
b=a
```



Computer memory

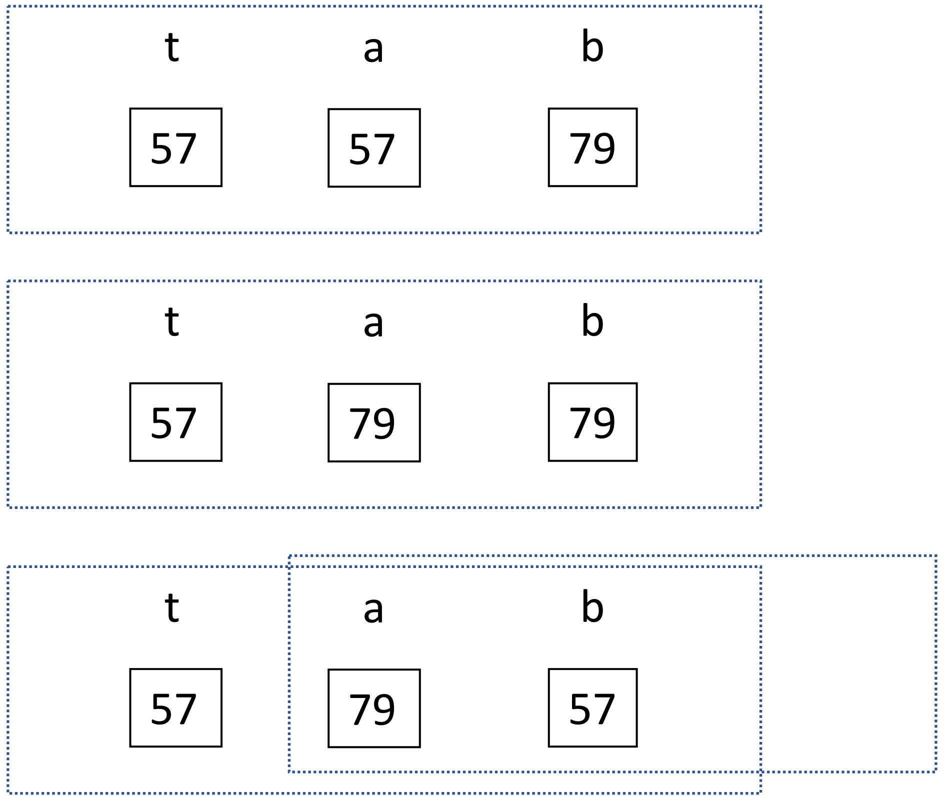
Algorithm:

t=a

a=b

b=t

Computer Memory



Target configuration

```
x = 16
y = 10
total = x+y
y = 11
```

Question!

What is total equal to after these instructions?

A) 26

B) 25

C) 37

D) 27

Basic Constructs for Algorithms

- Sequence
- Decision
- Iteration

Algorithm

An algorithm can be loosely defined in programming terms as a set of detailed, unambiguous and ordered instructions developed to describe the processes necessary to produce the desired output from the given input.

Important structures

- Sequence
- Decision
- Loop (iteration)

Pseudocode

- Statements are written in simple English.
- Each instruction is written on a separate line.
- Keywords and indentation are used to signify particular control structures.
- Each set of instructions is written from top to bottom with only one entry and one exit.
- Groups of statements may be formed into modules, and that group given a name.

Sequential statements

- Basic structure of straightforward algorithms, e.g. directions, recipes

```
Statement a
Statement b
Statement c
Statement d etc…
```

Input and Output

- Input – a computer can receive information from a user or file
- Output – A computer can write information to the user or a file

Assignments

We can create variables and assign values

a = 5

B = 6

c = a + b

- The variable on the left is given or ‘assigned’ the value on the right.
- The expression on the right must be valid and result in data of a suitable type

Arithmetic operations

Most languages have arithmetic operators + (add), - (subtract), * (multiply), / (divide)

e.g. a = (5+x)*3/y

If Statement

```
If condition a is true
        statement(s) in true case
End If
```

- Condition is a Boolean condition (true or false)

<	less than
>	greater than
==	equal to
<=	less than or equal
>=	greater than or equal to

```
IF age > 67
        fare = 0
ENDIF
```

If Else Statement

```
IF condition a is true
        statement(s) in true case
ELSE
        statement(s) in false case
ENDIF
```

```
IF age > 67
        fare = 0
ELSE
        fare = 5
ENDIF
```

Nested If Statements

```
IF record-code == 'A' THEN
        counterA = counterA + 1
ELSE
        IF record-code == 'B’
                counterB = counterB + 1
        ELSE
                IF record-code == 'C’
                        counterC = counterC + 1
                ELSE
                        error-counter = error-counter + 1
                ENDIF
        ENDIF
ENDIF
```

If Elseif Else

```
If condition a is true
        statement(s) if a is true
ELSEIF condition b is true
        statement(s) if a is false and b is true
ELSE
        statement(s) if none of the above conditions are true
ENDIF
```

```
IF record-code == 'A' THEN
        counterA = counterA + 1
ELSEIF record-code == 'B’
        counterB = counterB + 1
ELSEIF record-code == ‘C’
        counterC = counterC + 1
ELSE
        error-counter = error-counter + 1
ENDIF
```

Loops

- While loop
  - When there is a condition for terminating
- For loop
  - when a set number of iterations are required

While loop

```
WHILE condition a is true
        statements to execute while condition is true
ENDWHILE

e.g.
Total = 0
WHILE input != 0
        ask for user input
        store input in x
        Total = Total +x
ENDWHILE
```

For Loop

```
For - counter initialization; terminating condition; counter increment
        statements to execute while condition is true
End For
```

e.g.

```
total = 0
FOR i =0;i<10; i = i+1
        total = total + i
ENDFOR
```

```
i = 0
total = 0
WHILE i < 10
    total = total + i
    i = i + 1
ENDWHILE
```

Summary

- Sequential statements
  - Input and Output
  - Variable creation and assignment
  - Arithmetic operations
- Decisions
  - IF statements
- Iterations
  - WHILE and FOR loops