Introduction and Pseudocode

1 Algorithms

Algorithm - A method or a process followed to solve a problem
Properties an algorithm must have

- Correctness
- Composed of concrete unambiguous steps
- The number of steps must be finite
- Must terminate

2 Data Structures

Data Structure - A particular way of storing and organizing data in a computer so it can be used efficiently

3 Machine Model

Random Access Machine

- Memory consists of an infinite array
- Instructions executed sequentially one at a time
- All instructions take unit time. Running time is the number of executions executed

4 Pseudocode

To describe algorithms we will use generic pseudocode, not any one programming language.
There are no very strict rules on how it should be written.

Will often use variables: int, float, char, str
Declare variable type before using.

Different formats for setting equal.

Some conventions declare type separate to value, some do not

Logical operators included.

Keyword for output is print, to concatenate strings use a comma, for example ”The value of z is”,z

4.1 If then else

```
if condition then
    statement
end if
```

To make one thing happen if a statement is true, and one if it is false, two if statements can be used, but using an else statement is better, for example

```
if x ≠ 0 then
    z=y/x
else
    print "division by zero error"
end if
```

end if and indentation is used to make things clearer

4.2 For loop

If you want to iterate some (numeric) variable through some range

Great many variations in how languages do this, simplest is probably

```
for variable=lower to upper do
    body
end for
```

Body is simply a sequence of statements
Example:

```
s=0
integer L
integer U
for i=L to U do
    s=s+i
end for
print s
```

Things this for loop does without being written down:

- Set I=L
- Iterate i by 1
- Check if $i \leqslant U$

In this for loop, it can only iterate consecutive integers
A more generic one can iterate over a given bases set:

```
for value in {value1, value2 ...} do
    body
end for
```

To iterate in more than 1, for example in 2, this can also use negative numbers to reduce:

```
for i=0 to 9;i+=2 do
    print i
end for
```

4.3 While loop

Do something while a condition is true:

```
while condition do
    body
end while
```

Important difference between **for** and **while** over numerical values:
for increments loop-variable automatically; **while** doesn't

Everything that happens in a **while** loop is explicit, unlike a for loop

```
for i=1 to 10 do
    print i
end for
```

```
i=1
while $i \leqslant 10$ do
    print i
    i=i+1
end while
```