Pseudocode

- So the general structure of all programs is:

```
PROGRAM <ProgramName>:
<Do stuff>
END.
```

SEQUENCE

Pseudocode

- When we write programs, we assume that the computer executes the program starting at the beginning and working its way to the end.
- This is a basic assumption of all algorithm design.

Pseudocode

- When we write programs, we assume that the computer executes the program starting at the beginning and working its way to the end.
- This is a basic assumption of all algorithm design.
- We call this SEQUENCE.

Pseudocode

- In Pseudo code it looks like this:

```
Statement1;
Statement2;
Statement3;
Statement4;
Statement5;
Statement6;
Statement7;
Statement8;
```

Pseudocode

- For example, for making a cup of tea:

```
Organise everything together;
Plug in kettle;
Put teabag in cup;
Put water into kettle;
Wait for kettle to boil;
Add water to cup;
Remove teabag with spoon/fork;
Add milk and/or sugar;
Serve;
```

Pseudocode

- Or as a program:

```
PROGRAM MakeACupOfTea:
 Organise everything together;
 Plug in kettle;
 Put teabag in cup;
 Put water into kettle;
 Wait for kettle to boil;
 Add water to cup;
 Remove teabag with spoon/fork;
 Add milk and/or sugar;
 Serve;
END.
```

Pseudocode

- Or as a program:

```
PROGRAM MakeACupOfTea:
 Organise everything together;
 Plug in kettle;
 Put teabag in cup;
 Put water into kettle;
 Wait for kettle to boil;
 Add water to cup;
 Remove teabag with spoon/fork;
 Add milk and/or sugar;
 Serve;
END.
```

SELECTION

Pseudocode

- What if we want to make a choice, for example, do we want to add sugar or not to the tea?

Pseudocode

- What if we want to make a choice, for example, do we want to add sugar or not to the tea?
- We call this SELECTION.

Pseudocode

- So, we could state this as:

```
IF (sugar is required)
    THEN add sugar;
    ELSE don't add sugar;
ENDIF;
```

Pseudocode

- Or, in general:

```
IF (<CONDITION>)
    THEN <Statements>;
    ELSE <Statements>;
ENDIF;
```

Pseudocode

- Or to check which number is biggest:

```
IF (A > B)
    THEN Print A + “is bigger”;
    ELSE Print B + “is bigger”;
ENDIF;
```

Pseudocode

- Adding a selection statement in the program:

```
PROGRAM MakeACupOfTea:
 Organise everything together;
 Plug in kettle;
 Put teabag in cup;
 Put water into kettle;
 Wait for kettle to boil;
 Add water to cup;
 Remove teabag with spoon/fork;
 Add milk;
 IF (sugar is required)
   THEN add sugar;
   ELSE do nothing;
 ENDIF;
 Serve;
END.
```

Pseudocode

- Adding a selection statement in the program:

```
PROGRAM MakeACupOfTea:
 Organise everything together;
 Plug in kettle;
 Put teabag in cup;
 Put water into kettle;
 Wait for kettle to boil;
 Add water to cup;
 Remove teabag with spoon/fork;
 Add milk;
 IF (sugar is required)
   THEN add sugar;
   ELSE do nothing;
 ENDIF;
 Serve;
END.
```

ITERATION

Pseudocode

- What if we need to tell the computer to keep doing something until some condition occurs?

Pseudocode

- What if we need to tell the computer to keep doing something until some condition occurs?
- Let’s say we wish to indicate that the you need to keep filling the kettle with water until it is full.

Pseudocode

- What if we need to tell the computer to keep doing something until some condition occurs?
- Let's say we wish to indicate that the you need to keep filling the kettle with water until it is full.
- We need a loop, or ITERATION.

Pseudocode

- So, we could state this as:

```
WHILE (Kettle is not full)
    DO keep filling kettle;
ENDWHILE;
```

Pseudocode

- Or, in general:

```
WHILE (<CONDITION>)
    DO <Statements>;
ENDWHILE;
```

Pseudocode

- Or to print out the numbers 1 to 5:

```
A = 1;
WHILE(A < 5)
    DO Print A;
        A = A + 1;
ENDWHILE;
```

Pseudocode

- What is the benefit of using a loop?

Pseudocode

- Consider the problem of searching for an entry in a phone book with only condition:

Pseudocode

- Consider the problem of searching for an entry in a phone book with only condition:

```
Get first entry
If this is the required entry
  Then write down phone number
  Else get next entry
  If this is the correct entry
   then write done entry
   else get next entry
          if this is the correct entry
                  …….
```

Pseudocode

- This could take forever to specify.

Pseudocode

- This could take forever to specify.
- There must be a better way to do it.

Pseudocode

- We may rewrite this as follows:

```
Get first entry;
Call this entry N;
WHILE N is NOT the required entry
DO Get next entry;
     Call this entry N;
ENDWHILE;
```

Pseudocode

- We may rewrite this as follows:

```
Get first entry;
Call this entry N;
WHILE N is NOT the required entry
DO Get next entry;
     Call this entry N;
ENDWHILE;
```

- This is why we love loops!

Pseudocode

- Or as a program:

```
PROGRAM MakeACupOfTea:
 Organise everything together;
 Plug in kettle;
 Put teabag in cup;
 WHILE (Kettle is not full)
   DO keep filling kettle;
 ENDWHILE;
 Wait for kettle to boil;
 Add water to cup;
 Remove teabag with spoon/fork;
 Add milk;
 IF (sugar is required)
   THEN add sugar;
   ELSE do nothing;
 ENDIF;
 Serve;
END.
```

Pseudocode

- Or as a program:

```
PROGRAM MakeACupOfTea:
 Organise everything together;
 Plug in kettle;
 Put teabag in cup;
 WHILE (Kettle is not full)
   DO keep filling kettle;
 ENDWHILE;
 Wait for kettle to boil;
 Add water to cup;
 Remove teabag with spoon/fork;
 Add milk;
 IF (sugar is required)
   THEN add sugar;
   ELSE do nothing;
 ENDIF;
 Serve;
END.
```

EXAMPLES

Pseudocode

- So let's say we want to express the following algorithm:
  - *Read in a number and print it out.*

Pseudocode

```
PROGRAM PrintNumber:
    Read A;
    Print A;
END.
```

Pseudocode

- So let's say we want to express the following algorithm:
  - *Read in a number and print it out double the number.*

Pseudocode

```
PROGRAM PrintDoubleNumber:
    Read A;
    B = A*2;
    Print B;
END.
```

Pseudocode

- So let's say we want to express the following algorithm:
  - *Read in a number, check if it is odd or even.*

Pseudocode

```
PROGRAM IsOddOrEven:
    Read A;
    IF (A/2 gives a remainder)
            THEN Print "It's Odd";
            ELSE Print "It's Even";
    ENDIF;
END.
```

Pseudocode

- So let's say we want to express the following algorithm to print out the bigger of two numbers:
  - *Read in two numbers, call them A and B. Is A is bigger than B, print out A, otherwise print out B.*

Pseudocode

```
PROGRAM PrintBiggerOfTwo:
    Read A;
    Read B;
    IF (A>B)
            THEN Print A;
            ELSE Print B;
    ENDIF;
END.
```

Pseudocode

- So let's say we want to express the following algorithm to print out the bigger of three numbers:
  - *Read in three numbers, call them A, B and C.*
    - *If A is bigger than B, then if A is bigger than C, print out A, otherwise print out C.*
    - *If B is bigger than A, then if B is bigger than C, print out B, otherwise print out C.*

Pseudocode

```
PROGRAM BiggerOfThree:
    Read A;
    Read B;
    Read C;
    IF (A>B)
            THEN IF (A>C)
                                THEN Print A;
                                ELSE Print C;
                           END IF;
            ELSE IF (B>C)
                                THEN Print B;
                                ELSE Print C;
                       END IF;
    END IF;
END.
```

Pseudocode

- So let's say we want to express the following algorithm:
  - *Print out the numbers from 1 to 5*

Pseudocode

```
PROGRAM Print1to5:
    A = 1;
    WHILE (A != 6)
    DO  Print A;
        A = A + 1;
    ENDWHILE;
END.
```

Pseudocode

- So let's say we want to express the following algorithm:
  - *Add up the numbers 1 to 5 and print out the result*

Pseudocode

```
PROGRAM PrintSum1to5:
    Total = 0;
    A = 1;
    WHILE (A != 6)
    DO  Total = Total + A;
        A = A + 1;
    ENDWHILE;
    Print Total;
END.
```

Pseudocode

- So let's say we want to express the following algorithm:
  - *Read in a number and check if it's a prime number.*

Pseudocode

- So let's say we want to express the following algorithm:
  - *Read in a number and check if it's a prime number.*
  - *What's a prime number?*

Pseudocode

- So let's say we want to express the following algorithm:
  - *Read in a number and check if it's a prime number.*
  - *What's a prime number?*
  - *A number that's only divisible by itself and 1, e.g. 7.*

Pseudocode

- So let's say we want to express the following algorithm:
  - *Read in a number and check if it's a prime number.*
  - *What's a prime number?*
  - *A number that's only divisible by itself and 1, e.g. 7.*
  - *Or to put it another way, every number other than itself and 1 gives a remainder, e.g. For 7, if 6, 5, 4, 3, and 2 give a remainder then 7 is prime.*

Pseudocode

- So let's say we want to express the following algorithm:
  - *Read in a number and check if it's a prime number.*
  - *What's a prime number?*
  - *A number that's only divisible by itself and 1, e.g. 7.*
  - *Or to put it another way, every number other than itself and 1 gives a remainder, e.g. For 7, if 6, 5, 4, 3, and 2 give a remainder then 7 is prime.*
  - *So all we need to do is divide 7 by all numbers less than it but greater than one, and if any of them have no remainder, we know it's not prime.*

Pseudocode

- So,
- If the number is 7, as long as 6, 5, 4, 3, and 2 give a remainder, 7 is prime.
- If the number is 9, we know that 8, 7, 6, 5, and 4, all give remainders, but 3 does not give a remainder, it goes evenly into 9 so we can say 9 is not prime

Pseudocode

- So remember,
  - if the number is 7, as long as 6, 5, 4, 3, and 2 give a remainder, 7 is prime.
- So, in general,
  - if the number is A, as long as A-1, A-2, A-3, A-4, ... 2 give a remainder, A is prime.

Pseudocode

```
PROGRAM Prime:
    Read A;
    B = A - 1;
    IsPrime=True;
    WHILE (B != 1)
    DO  IF (A/B gives no remainder)
            THEN IsPrime= False;
        ENDIF;
        B = B - 1;
    ENDWHILE;
    IF (IsPrime == true)
        THEN Print "Prime";
        ELSE Print "Not Prime";
    ENDIF;
END.
```