

## String handling

Term	Definition
String	A string is a sequence of keyboard characters.
String handling	Manipulating data in a string. e.g. selecting first three characters.

The following conventions will be used for string handling.

Construct	Example usage
Variable names	myvariable
Define variable as string	myvariable is String
Length of string	len (string)
Mid string where x is the offset and y is the length	mid (string, start, length)
Replace part of a string	replace (string, find, replacewith)
Compare two strings	str(Comp (string1, string2))

To create a string, we need to define a variable as a string and assign a value to the variable.

```
greeting is String
greeting = "Hello from me!"
print greeting
```

**Hello from me!**

To find out how many characters are in a string, use the `len` command. Any keyboard stroke is counted as a character, so spaces and special characters are counted as well as letters and numbers.

```
greeting is String
greeting = "Hello from me!"
length = len(greeting)
print length
```

**13**

To discover the contents of all or part of a string, the `mid` command is used.

```
txt is String
partMessage is String
txt = "Have a happy birthday"
partMessage = mid(txt, 8, 5)
print partMessage
```

**happy**

The `replace` command is used to replace part of a string.

```
txt is String
message is String
txt = "Have a happy birthday"
message = replace(txt, "happy",
"fantastic")
print message
```

**Have a fantastic birthday**

The process of combining a string with text or combining two strings with or without additional text is called concatenation.

```
txt1 is String
txt2 is String
concatString is String
txt1 = "Sarah"
txt2 = "Smith"
print "Welcome " & txt1 & txt2
```

**Welcome Sarah Smith**

## Sorting and Searching

Term	Definition
Sorting	In computer science, arranging in an ordered sequence is called sorting.
Searching	To examine data in a file to find items that match given criteria.

## Sorting

The two sorting algorithms required for this qualification are the bubble sort and the merge sort.

```
Declare BubbleSort(bubbleList):
    exchanges = True
    passnum = len(bubbleList)-1
    while passnum > 0 and exchanges = True
        exchanges = False
        for i = 1 to n
            if bubbleList[i]>bubbleList[i+1]:
                exchanges = True
                temp = bubbleList[i]
                bubbleList[i] = bubbleList[i+1]
                bubbleList[i+1] = temp
            end if
        next i
        passnum = passnum-1
    end while
end Subroutine

bubbleList=[20,30,40,90,50,60,70,80,100,110]

BubbleSort (bubbleList)
print (bubbleList)
```

## Searching

The two searching algorithms required for this qualification are the linear and the binary search. The linear search is a very simple search algorithm. Each item in the data set is compared with the search condition in sequence until the item is found or the end of the data set is reached.

```
Declare linearSearch(dataList, searchItem)

position = 0
found = false

while position < len(dataList) and found = false
    if dataList[position] = searchItem then
        found = true
    else
        position = position + 1
    end if
end while

testList = [1,3,21,45,57,17,34,65]
linearSearch(testList, 45)
linearSearch(testList, 20)
```

Details of the merge sort and the binary search are included in the WJEC Notes.