Structured Programming

- Parameters act as input.

Header of Function/ Procedure

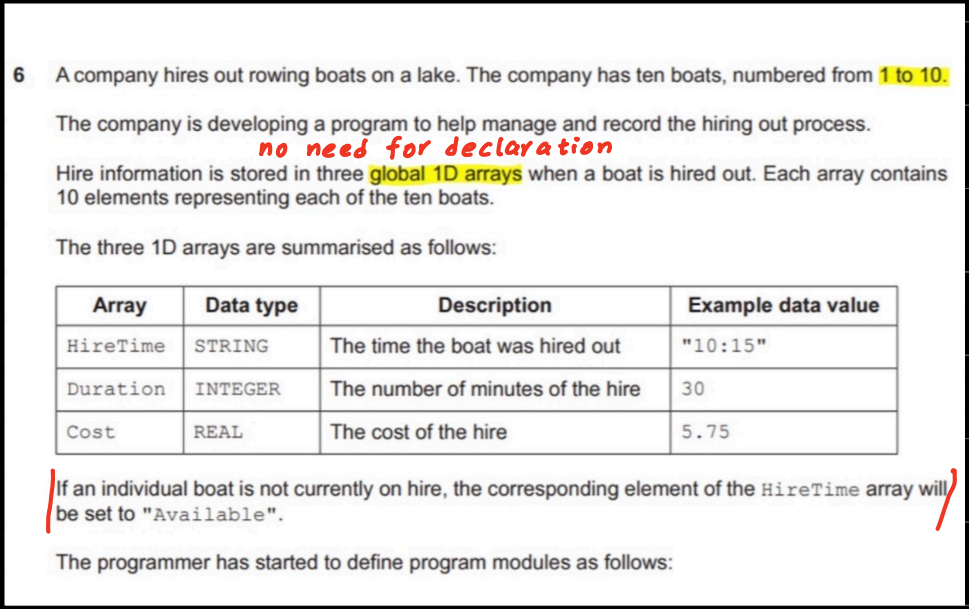
```
                                                      (Header)
FUNCTION Name( Name of variable : data type) RETURN Data type

END FUNCTION (Footer)

PROCEDURE Name( Name of variable : data type) (Header)

END PROCEDURE (Footer)
```

Q- Write a function that take x as a parameter and return its square.

```
FUNCTION SQUARE (ByVal. x : INTEGER) RETURN INTEGER
         Sq = x * x
         Return Sq
END FUNCTION
```

- No need to take input seperately

Functions and Procedures With Arrays



6 A company hires out rowing boats on a lake. The company has ten boats, numbered from 1 to 10.

The company is developing a program to help manage and record the hiring out process.

no need for declaration

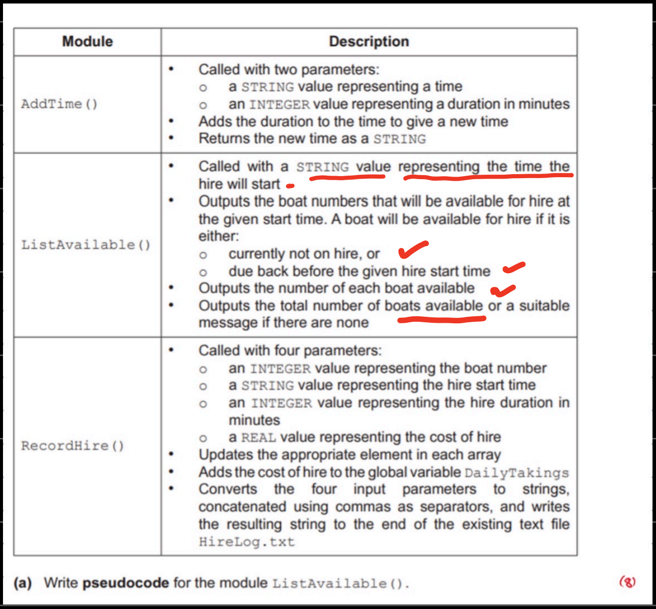
Hire information is stored in three global 1D arrays when a boat is hired out. Each array contains 10 elements representing each of the ten boats.

The three 1D arrays are summarised as follows:

Array	Data type	Description	Example data value
HireTime	STRING	The time the boat was hired out	"10:15"
Duration	INTEGER	The number of minutes of the hire	30
Cost	REAL	The cost of the hire	5.75

If an individual boat is not currently on hire, the corresponding element of the HireTime array will be set to "Available".

The programmer has started to define program modules as follows:



Module	Description
AddTime()	• Called with two parameters: o a STRING value representing a time o an INTEGER value representing a duration in minutes • Adds the duration to the time to give a new time • Returns the new time as a STRING
ListAvailable()	• Called with a STRING value representing the time the hire will start • Outputs the boat numbers that will be available for hire at the given start time. A boat will be available for hire if it is either: o currently not on hire, or ✓ o due back before the given hire start time ✓ • Outputs the number of each boat available ✓ • Outputs the total number of boats available or a suitable message if there are none
RecordHire()	• Called with four parameters: o an INTEGER value representing the boat number o a STRING value representing the hire start time o an INTEGER value representing the hire duration in minutes o a REAL value representing the cost of hire • Updates the appropriate element in each array • Adds the cost of hire to the global variable DailyTakings • Converts the four input parameters to strings, concatenated using commas as separators, and writes the resulting string to the end of the existing text file HireLog.txt

(a) Write **pseudocode** for the module ListAvailable(). (8)

```
PROCEDURE ( Hirestarttime : STRING)
     DECLARE Boatnumavailable, Index : INTEGER

     Boatnumavailable = 0
     FOR Index = 1 TO 10
          IF Hiretime[Index] = “Available”
            THEN
               Boatnumavailable= Boatnumavailable + 1
               OUTPUT “Boat”, Index, “is available”
            ELSE
               IF Hirestarttime > Addtime(Hiretime[Index], Duration[Index])
                   THEN
                      Boatnumavailable = Boatnumavailable +1
                      OUTPUT “Boat”, Index, “is available”
               ENDIF
          ENDIF
          IF Boatnumavailable >0
```

- Total number of boats.
- Each boat.

```
                THEN
                    OUTPUT "There are ", Boatnumavailable, " boats available"
                ELSE
                    OUTPUT "Sorry, no boats available"
            ENDIF
        ENDFOR
END PROCEDURE
```

Local Variable: Variable declared within a module. Can not be used in other modules

Global Variable: A variable that is accessible from all modules. Can be used in all modules.

Parameter: A variable applied to a function/procedure that allows one to pass in a value for the functions/procedures to use.

Arguement: The value passed to a procedure / Function

- Procedure calls are single stand-alone statements whereas functions form part of an expression on the right hand side.

- The RETURN value in a function replaces the call.

- **Header:** The first statement in the definition of a procedure or function which contains names, parameters and for a funtion returned value's data type.

- Values are passed either by reference or by value only to procedures

Return value: The value which replaces the function call.

Procedure / function interface: Communication b/w procedures, functions and the program in which they are called.

- when functions / procedures are defined with parameters, the parameters must match those in the function definition and take care of the order in which parameters are passed.