Chapter 19: Recursion

JUN 2020 9608/41 (Q2)

(a) Programs can be written using recursion.

Tick (✓) one or more boxes to show the features that **must** be included in a valid recursive algorithm.

Feature	Must be included
Incrementation	
General case	
Base case	
Selection case	
It calls itself	

[2]

(b) The following recursive procedure outputs every even number from the positive parameter value down to and including 2.

The procedure checks if the integer parameter is an even or an odd number. If the number is odd, the procedure converts it to an even number by subtracting 1 from it.

The function `MOD(ThisNum : INTEGER, ThisDiv : INTEGER)` returns the remainder value when `ThisNum` is divided by `ThisDiv`.

Complete the **pseudocode** for the recursive procedure.

```
PROCEDURE Count(BYVALUE ..... : INTEGER)
    IF .....(Number, 2) <> 0
        THEN
            Number ← Number - 1
    ENDIF
    OUTPUT .....
    IF Number > 0
        THEN
            .....(..... - 1)
    ENDIF
ENDPROCEDURE
```

[5]

(c) A program allows guests to input a meal option at a wedding.

Guests can choose meal option 1 or meal option 2.

The program will keep count of the numbers of each meal option chosen.

The program ends when a value other than 1 or 2 is entered. It then outputs the count of each meal option.

```
PROCEDURE MealsCount(BYREF MealOption1 : INTEGER, MealOption2 : INTEGER)
    DECLARE MealOption : INTEGER
    DECLARE MoreMeals : BOOLEAN
    MoreMeals ← True
    WHILE MoreMeals = True
        INPUT MealOption
        IF MealOption = 1
            THEN
                MealOption1 ← MealOption1 + 1
            ELSE
                IF MealOption = 2
                    THEN
                        MealOption2 ← MealOption2 + 1
                    ELSE
                        OUTPUT MealOption1, " ", MealOption2
                        MoreMeals ← False
                ENDIF
        ENDIF
    ENDWHILE
ENDPROCEDURE
```

The program contains a conditional loop.

Use **pseudocode** to rewrite the conditional loop as a recursive algorithm.

```
PROCEDURE MealsCount(BYREF MealOption1 : INTEGER, MealOption2 : INTEGER)

    DECLARE MealOption : INTEGER
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

```
ENDPROCEDURE
```

[5]

JUN 2020 9608/42 (Q3)

Recursive algorithms can be used when creating programs.

(a) Describe what is meant by a **recursive algorithm**.

.....

.....

.....

.....

.....

..... [3]

(b) A string that is a palindrome, reads the same forwards as it does backwards. For example, the name Anna is a palindrome.

The function `Substring(Variable, StartingCharacter, NumberOfCharacters)` returns one or more characters from a string. The first character is at position 0.

For example, the string `"Happy"` is stored in the variable `Word`.

- `Substring(Word, 1, 1)` would return the character `"a"`.
- `Substring(Word, 2, 3)` would return the characters `"ppy"`.

The function `Length()` returns the length of the string as an integer. For example, `Length(Word)` returns `5`.

The following is a recursive function to find out whether a string is a palindrome. The function returns `True` if the parameter is a palindrome, and returns `False` if it is not a palindrome.

Complete the **pseudocode** for the recursive algorithm to indicate whether a string is a palindrome.

```
FUNCTION IsPalindrome(CheckWord : STRING) RETURNS BOOLEAN
    IF ..... <= 1
        THEN
            RETURN .....
    ENDIF
    IF Substring(CheckWord, 0, 1) <>
             Substring(CheckWord, ..... (CheckWord)-1, 1)
        THEN
            RETURN .....
        ELSE
            RETURN .....(Substring(CheckWord, 1,
                                   Length(CheckWord)-2))
    ENDIF
ENDFUNCTION
```

[5]

(c) The function `FindPower()` is a recursive function that calculates the result of a base number to the power of the exponent. For example, the result of $2^4 = 16$, as 2*2*2*2 = 16. In this example, 2 is the base number and 4 is the exponent.

The base number and the exponent are passed as parameters.

Write **pseudocode** for the recursive function `FindPower()`. Assume both the base and the exponent are positive integers.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

..... [5]

JUN 2020 9608/41 (Q2) - ANSWERS

2(a)

1 mark for the first 3 rows
1 mark for the last 2 rows

Feature	Must be included
Incrementation	
General case	✓
Base case	✓
Selection case	
It calls itself	✓

2(b)

1 mark for each of the spaces filled in

```
PROCEDURE Count(BYVALUE Number : INTEGER)
    IF MOD(Number, 2) <> 0
        THEN
            Number ← Number - 1
    ENDIF
    OUTPUT Number
    IF Number > 0
        THEN
            CALL Count(Number - 1)
    ENDIF
ENDPROCEDURE
```

2(c)

1 mark per bullet point

- Recursive call
- ..with correct parameters
- …in correct working place(s)
- Removal of loop
- Remainder of program in correct place

```
PROCEDURE MealsCount(BYREF MealOption1 : INTEGER, MealOption2 : INTEGER)
    DECLARE MealOption : INTEGER
    INPUT MealOption
    IF MealOption = 1
        THEN
            MealOption1 ← MealOption1 + 1
            CALL MealsCount(MealOption1, MealOption2)
        ELSE
          IF MealOption = 2
            THEN
              MealOption2 ← MealOption2 + 1
              CALL MealsCount(MealOption1, MealOption2)
            ELSE
              OUTPUT MealOption1, " ", MealOption2
          ENDIF
    ENDIF
ENDPROCEDURE
```

JUN 2020 9608/42 (Q3) - ANSWERS

3(a)	1 mark per bullet point • Have a base case // stopping condition • Work toward the base case // general case // changes state • Call itself // defined in terms of itself

3(b)	1 mark for each (shown in **bold**)
	
	`FUNCTION IsPalindrome(CheckWord : STRING) RETURNS BOOLEAN`
	`  IF Length(CheckWord) <= 1`
	`  THEN`
	`  RETURN True`
	`  ENDIF`
	`  IF Substring(CheckWord, 0, 1) <>`
	`  Substring(CheckWord,Length(CheckWord)-1, 1)`
	`  THEN`
	`  RETURN False`
	`  ELSE`
	`  RETURN IsPalindrome(Substring(CheckWord,1,`
	`  Length(CheckWord)-2))`
	`  ENDIF`
	`ENDFUNCTION`

3(c)	**1 mark** per bullet point
	
	• Function is defined and returns integer with correct parameters
	• Returns 1 if exponent is 0 // returns base if exponent is 1
	• Calculates base to power with recursive call …
	• … with correct parameters
	• Returns result of calculation
	e.g.
	`FUNCTION FindPower(Base:INTEGER, Exp: INTEGER) RETURNS INTEGER`
	`  DECLARE Value : INTEGER`
	`  IF Exp = 0`
	`  THEN`
	`  RETURN 1`
	`  ELSE`
	`  Value ← Base * FindPower(Base, Exp - 1)`
	`  RETURN Value`
	`  ENDIF`
	`ENDFUNCTION`

JUN 2022 9618/42 (Q2ci)

(c) The following pseudocode function uses recursion to perform a binary search in the first row of the array, for the value `SearchValue` in the array `SearchArray`.

The function returns `-1` if the item was not found, or it returns the index where it is found.

There are **six** incomplete statements.

```
FUNCTION BinarySearch(SearchArray, Lower, Upper, SearchValue)RETURNS
                                                              INTEGER
  IF Upper >= Lower THEN
    Mid ← (Lower + (Upper - 1)) DIV .....
    IF SearchArray[0, Mid] = ..... THEN
      RETURN .....
    ELSE
      IF SearchArray[0, Mid] > SearchValue THEN
        RETURN BinarySearch(SearchArray, ....., Mid - 1,
                                                  SearchValue)
      ELSE
        RETURN BinarySearch(SearchArray, Mid + 1, .....,
                                                  SearchValue)
      ENDIF
    ENDIF
  ENDIF
  RETURN .....
ENDFUNCTION
```

Note: the arithmetic operator `DIV` performs integer division, e.g. the result of `10 DIV 3` will be `3`.

(i) Write program code for the recursive function `BinarySearch()`.

JUN 2021 9618/41 (Q1)

Study the following pseudocode for a recursive function.

```
FUNCTION Unknown(BYVAL X, BYVAL Y : INTEGER) RETURNS INTEGER
  IF X < Y THEN
    OUTPUT X + Y
    RETURN (Unknown(X + 1, Y) * 2)
  ELSE
    IF X = Y THEN
      RETURN 1
    ELSE
      OUTPUT X + Y
      RETURN (Unknown(X - 1, Y) DIV 2)
    ENDIF
  ENDIF
ENDFUNCTION
```

The operator `DIV` returns the integer value after division e.g. `13 DIV 2` would give `6`

(a) Write program code to declare the function `Unknown()`.

(b) The main program needs to run all **three** of the following function calls and output the result of each call:

```
Unknown(10, 15)
Unknown(10, 10)
Unknown(15, 10)
```

(i) For each of the **three** function calls, the main program needs to:

- output the value of the two parameters
- call the function with those parameters
- output the return value.

Write the program code for the main program.

(c) Rewrite the function `Unknown()` as an iterative function, `IterativeUnknown()`.

(d) The iterative function needs to be called **three** times with the same parameters as in **part (b)**.

(i) For each of the **three** function calls, the main program needs to:

- output the value of the two parameters
- call the iterative function with those parameters
- output the return value.

Amend the main program to perform these tasks.

NOV 2020 9608/41 (Q8)

8 Recursion can be used when writing computer programs.

Consider the following pseudocode algorithm.

```
01 FUNCTION NumberPattern(Value1, Value2, EndValue : INTEGER) RETURNS INTEGER
02    OUTPUT Value1
03    IF Value1 <= EndValue
04       THEN
05          Temp ← Value2
06          Value2 ← Value1
07          Value1 ← Value1 + Temp
08          RETURN NumberPattern(Value1, Value2, EndValue) + 1
09       ELSE
10          RETURN 0
11    ENDIF
12 ENDFUNCTION
```

(a) State the line number in the pseudocode algorithm that shows function `NumberPattern()` is recursive. Justify your choice.

Line number .....

Justification .....

.....

.....

[2]

(b) The function is called as follows:

```
NumberPattern(1,1,12)
```

Dry run the algorithm and complete the following trace table. State the final value returned.

Show your working.

Value1	Value2	Temp	EndValue	OUTPUT	RETURN value

Final value returned .....

(c) State the purpose of the algorithm.

.....

..... [1]

JUN 2022 9618/42 (Q2ci) - ANSWERS

VB.NET

```
Function BinarySearch(ByVal SearchArray(,) As Integer, Lower As Integer, Upper As Integer,
SearchValue As Integer)
  Dim Mid As Integer
  If Upper >= 0 Then
    Mid = (Lower + (Upper - 1)) \ 2
    If SearchArray(0, Mid) = SearchValue Then
      Return Mid
    ElseIf SearchArray(0, Mid) > SearchValue Then
      Return BinarySearch(SearchArray, Lower, Mid - 1, SearchValue)
    Else
      Return BinarySearch(SearchArray, Mid + 1, Upper, SearchValue)
    End If
  End If
  Return -1
End Function
```

JUN 2021 9618/41 (Q1) - ANSWERS

1(a)

```
Function Unknown(X, Y)
  If X < Y Then
    Console.WriteLine(X + Y)
    Return Unknown(X + 1, Y) * 2
  ElseIf X = Y Then
    Return 1
  Else
    Console.WriteLine(X + Y)
    Return Unknown(X - 1, Y) \ 2
  End If
End Function
```

1(b)

```
Console.WriteLine("10 and 15")
Console.WriteLine(Unknown(10, 15))
Console.WriteLine("10, 10")
Console.WriteLine(Unknown(10, 10))
Console.WriteLine("15, 10")
Console.WriteLine(Unknown(15, 10))
```

1(c)

```
Function IterativeUnknown(X, Y)
  Dim Total As Integer = 1
  While X <> Y
    Console.WriteLine(X + Y)
    If X < Y Then
      X = X + 1
      Total = Total * 2
    Else
      X = X - 1
      Total = Total \ 2
    End If
  End While
  Return Total
End Function
```

1(d)

```
Console.WriteLine("10 and 15")
Console.WriteLine(IterativeUnknown(10, 15))
Console.WriteLine("10, 10")
Console.WriteLine(IterativeUnknown(10, 10))
Console.WriteLine("15, 10")
Console.WriteLine(IterativeUnknown(15, 10))
```

NOV 2020 9608/41 (Q8) – ANSWERS

8(a)

- 8 …
- …it calls itself

8(b)

1 mark each:

- Final return value = 5
- Output column
- Return value column
- Value 1 and Value 2 columns
- Temp column

Value1	Value2	Temp	EndValue	OUTPUT	Return Value
1	1	1	12	1	5
2				2	4
3	2	2		3	3
5	3	3		5	2
8	5	5		8	1
13	8			13	0

8(c)

To output/find a value that is the addition of the two previous values // (output) Fibonacci sequence