User Defined Data Types

Q- What is meant by user defined data type?

- Derived from one or more existing data types
- used to extend built-in data types
- To meet programmer's requirement

Q- Explain why user defined data types are necessary?

- No suitable data type is provided by the language used.
- The programmer needs to specify a new data type
- that meets program's requirements

Data Type

Non-composite data type: Single data type that does not refer to another data type. e.g: Enumerated, Pointer, Real, String, Char, boolean, Integer

Composite data type: Single data type constructed from other data types

Composite examples :

Record: Collection of related items which may have different data types

List: Indexed collection that can have different data types

Set: Supports mathematical operations

Array: Collection of items with same data type

Class: Gives properties and method for an object

Queue

Linked list

Dictionary

Stack

Enumerated Data Type

- Non-composite data type
- Defined by a given list of all possible values
- in an order

Example: A B C D ...

1 2 3 ...

Mon Tue Wed...

Jan Feb Mar...

Pseudocode

```
TYPE
    DECLARE Name of Dt = ( _ , _ , _ )
END TYPE
```

Example

```
TYPE
    DECLARE Months = ( January, February , March , April,
                       May, August)
END TYPE
```

Q- Declare a variable currentmonth of data type 'months'

- DECLARE currentmonth : Months

Q. Assign 'August' in currentmonth

- Currentmonth ← August

Q- Declare variable 'previous month' of data type Months.

- DECLARE previousmonth : Months

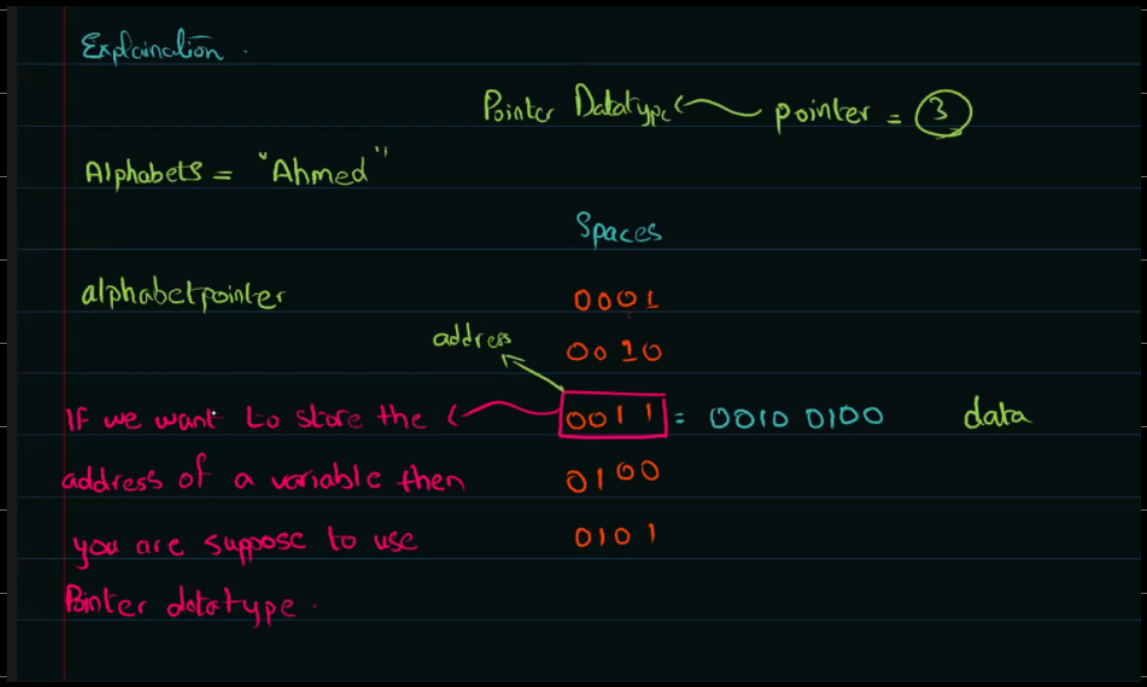
Q- Print previous month

- Previousmonth ← Currentmonth -1

  OUTPUT previousmonth

Pointer DataType

- Non-Composite Data type
- Used to reference a memory location



^ : This symbol represents pointer

@ : This symbol represents that the address is required not the value.

Pseudocode

```
TYPE
        Name of pointer data type = ^base data type
END TYPE
```

E.g:

```
        Number= 100
```

=> TYPE

```
        integer_pointer = ^INTEGER
END TYPE
```

Q- Declare a variable in which you store the address of integer variables.

```
·DECLARE Pointer: integer_pointer
```

Q- Store the address of Number1 into variable "Pointer"

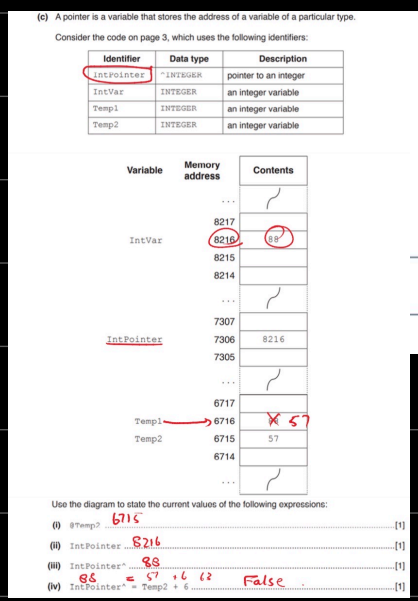
- Number1 = 100

- Pointer ← @ Number1

Dereferencing

- You have the address and you want the value on that address.

E.g: Pointer^ will return the value stored in Number1



(c) A pointer is a variable that stores the address of a variable of a particular type.

Consider the code on page 3, which uses the following identifiers:

Identifier	Data type	Description
IntPointer	^INTEGER	pointer to an integer
IntVar	INTEGER	an integer variable
Temp1	INTEGER	an integer variable
Temp2	INTEGER	an integer variable

Variable	Memory address	Contents
	...	
	8217	
IntVar	8216	88
	8215	
	8214	
	...	
	7307	
IntPointer	7306	8216
	7305	
	...	
	6717	
Temp1	6716	57
Temp2	6715	57
	6714	
	...	

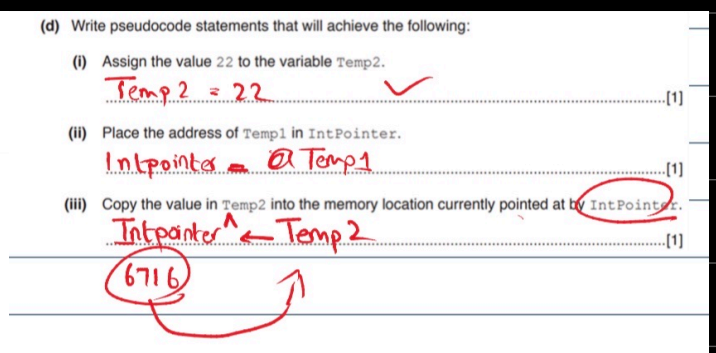
Use the diagram to state the current values of the following expressions:

(i) @Temp2 ..... [1]

(ii) IntPointer ..... [1]

(iii) IntPointer^ ..... [1]

(iv) IntPointer^ = Temp2 + 6 ..... [1]



(d) Write pseudocode statements that will achieve the following:

(i) Assign the value 22 to the variable Temp2.

Temp2 = 22 ..... [1]

(ii) Place the address of Temp1 in IntPointer.

IntPointer = @Temp1 ..... [1]

(iii) Copy the value in Temp2 into the memory location currently pointed at by IntPointer.

IntPointer^ ← Temp2 ..... [1]

6716

Record Data Type

- Composite Data type
- Group of multiple data type

Pseudocode

```
=>   TYPE   Name of Data type
             DECLARE Value 1 : Data type
             DECLARE Value 2 : Data type
     END TYPE
```

E.g:

```
     TYPE   Book
            DECLARE ISBN : INTEGER
            DECLARE Title: String
            DECLARE Genre: String
     END TYPE
```

Q- Write details of Book 1

```
DECLARE Book1 : Book
Book1.ISBN ← 123456
Book1.Title ← "Papersdock"
Book1.Genre ← "Horror"
```

- Stored like an array but data can be of same or different data type

Declaring a Range

Q- Declare a variable named number which contains "0" till "99" numbers

```
DECLARE Number: 0..99
```

String → `DECLARE Name : ("Monkey", "Bat", "Witch")`

Specific Datatype In Array

- DECLARE : ARRAY [1 : 10] OF STRING
  - → 3 specific string values ("Muhammad","Umar","Sami")

- DECLARE : ARRAY [1 : 3] OF (" Muhammad " , " Umar " , " Sami ")

- Note: IF there are more than one thing to store then declare array
- Average is always a real value.
- Whenever declaring specific data type, the variable will only be able to take up the specified values