

13 User-defined data types – Part 1

Name: _____ Class: _____ Date: _____

Vocabulary

Write each word three times.

English	Write it 3 times (English)
user-defined types	_____
enumerated type	_____
pointer	_____
dereference	_____

Recall

The built-in types (INTEGER, STRING, BOOLEAN...) cover simple cases. For richer problems you can make your own **user-defined types** .

New ideas

Read these first. Each idea has a short worked example. Then do the Practice.

■ 1 Enumerated type

TYPE Vehicle =



a variable may hold

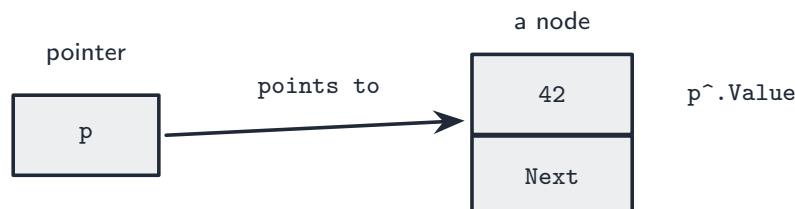
only one of these

An **enumerated type** is a fixed list of named values. A variable of that type can hold *only* one of them (good for days of the week, status codes).

Worked example.

```
TYPE Day = (Mon, Tue, Wed, Thu, Fri, Sat, Sun)
DECLARE today : Day
today ← Wed      // allowed: Wed is in the list
today ← Funday   // rejected: not one of the values
```

■ 2 Pointer



A **pointer** holds the memory address of another variable. To **dereference** it ($p^{\wedge}$) means to reach the variable it points to.

Worked example. Say a node holding 42 lives at address 2000. A pointer p stores 2000 (the address, not the 42). Writing $p^{\wedge}$ follows the address to reach the 42 itself.

Watch out: an enumerated value like `Wed` is a named constant you chose – not the string `"Wed"`, so write it without quotes. And a pointer stores an *address*, not the value it points to.

Practice

Try these in order.

13.1 Why might you use a *user-defined type*? [1]

13.2 What is an *enumerated type*? [1]

13.3 Give one good use of an enumerated type. [1]

13.4 What does a *pointer* hold? [1]

13.5 What does it mean to *dereference* a pointer? [1]

13.6 A program stores a traffic light's state. Explain why an enumerated type (`Red`, `Amber`, `Green`) is safer than using a `STRING`. [2]

13.7 Challenge. Write a short pseudocode declaration of an enumerated type `Suit` for the four suits in a deck of cards, then a variable of that type set to one value. [2]
