11. Programming

Starter Quiz · 课前小测

A-Level Computer Science Name: _____ Score: _____

1. A constant holds a fixed value that never changes while the program runs, whereas a variable can be reassigned.

   ☐ True ☐ False

2. "A mark of 50 or more passes" is written as 'IF Mark _____ 50 THEN'.

   _____

3. The key difference between a procedure and a function is that a function:

   ☐ returns a value that can be used in an expression

   ☐ cannot take parameters

   ☐ never does any work

   ☐ must be global

4. Which are benefits of using a constant for a value such as a tax rate? Select **all** that apply.

   ☐ it is set once and cannot be changed accidentally by the program

   ☐ a change is made in one place and reaches every use

   ☐ the identifier gives the value a meaningful name

   ☐ the program runs twice as fast

   Tick every correct option.

5. A CASE statement is cleaner than nested IFs when you are:

   ☐ testing one value against several options

   ☐ repeating a block of code

   ☐ reading a file

   ☐ declaring a constant

6. A function 'Square(x)' returns 'x * x'. What does the call 'Square(5)' return?

   _____

7. What is the value of '7 DIV 2'?

   _____

8. Which of these are valid guards in a Cambridge CASE statement? Select **all** that apply.

   ☐ '1:'

   ☐ '1, 2, 3:'

   ☐ '1 TO 5:'

   ☐ '> 200:'

   ☐ 'OTHERWISE:'

   Tick every correct option.

9. A good reason to write a subroutine is that:

   ☐ the same logic is needed in more than one place

   ☐ you want to use more global variables

   ☐ the program is only one line long

   ☐ you want to avoid naming things

10. What is the value of '7 MOD 2'?

    _____

11. Programming

Answers · 答案

A-Level Computer Science

1. **True**

   Constants (like Pi or a maximum score) make code clearer and let you change a recurring value in one place.

2. **>=**

   ”Or more” includes 50 itself, so the comparison is greater than or equal. ‘>‘ would fail a student on exactly 50.

3. **returns a value that can be used in an expression**

   A function returns a value (used in an expression); a procedure performs an action and returns nothing.

4. **it is set once and cannot be changed accidentally by the program; a change is made in one place and reaches every use; the identifier gives the value a meaningful name**

   The three mark-scheme benefits are safety, a single point of change, and readability. Speed is not one of them.

5. **testing one value against several options**

   CASE matches one value against many possibilities; deep nested IFs become hard to read.

6. **25**

   $5 \times 5 = 25$ — the value the function hands back to its caller (the frame popped off the call stack).

7. **3**

   DIV is integer division: $7 \div 2 = 3$ remainder 1, so DIV gives 3.

8. **‘1:‘; ‘1, 2, 3:‘; ‘1 TO 5:‘; ‘OTHERWISE:‘**

   A single value, a value list, a range and OTHERWISE. A comparison such as ‘> 200‘ is not a guard; anything not covered goes to OTHERWISE.

9. **the same logic is needed in more than one place**

   Subroutines remove duplication, give a named purpose, and can be tested in isolation.

10. **1**

    MOD gives the remainder: $7 \div 2$ leaves remainder 1.