R 50 Assignment, Variables, Constants & Sequences - Reading

Computer programs are normally written in **high level languages** that are close to how humans think rather than computers.

In computer programs we often want to store **values**. For instance, we may want to store a player's name or score in a game. The values that we store might need to change in the program so we store them in **variables** (as the values can *vary*).

A variable is an identifier (name) that points to a memory location in RAM that stores a value that can change when the program is run

The rules as to how we write computer code are known as **syntax**. Here we will use syntax that is not for a specific language but is easy to understand no matter what language you decide to actually program in.

Putting a value into a variable is known as **assignment**. If we do this when the variable is first set up, it is known as **initialisation**.

Syntax for assignment

```
variable name = value
```

Example of assignment

```
score = 17
```

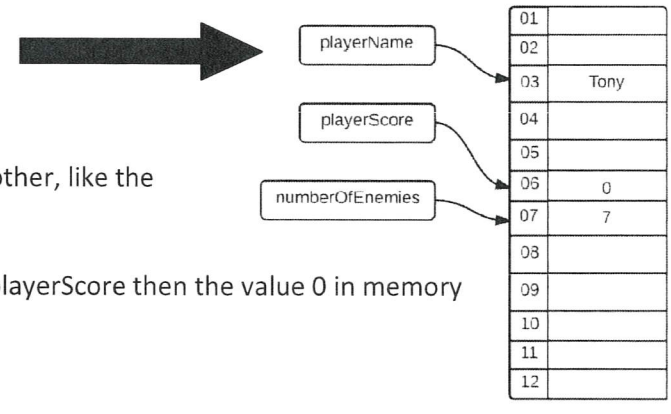
The = symbol is **NOT** an equals symbol. It is the **assignment operator** in this situation. For the above example we say that "the variable score is **assigned** the value 17".

In general, variables are written with no spaces and in lower case. The can be written with an underscore separating words, this is known as **snake case**. Alternatively, words can be joined with each word starting with a capital letter, this is known as **camel case**.

Example snake case variable names	Example camel case names
player_name	playerName
player_score	playerScore
number_of_enemies	numberOfEnemies

The following code will set up three variables. The variable names, pointers, memory locations and values in RAM are shown on the right as they would be at the end of the three lines of code running.

```
playerName = "Tony"
playerScore = 0
numberOfEnemies = 7
```



Lines of code which execute one after the other, like the above program, are known as a **sequence**.

If the following assignment were made to playerScore then the value 0 in memory would become replaced by 10.

```
playerScore = 10
```

If we want to store a value that doesn't change while the programming is running then we store it in a **constant**. Constants are normally written with capital letters. E.g. MAX_NUMBER_OF_PLAYERS

Q 50

1. Match the words on the left to their meanings on the right.

variable	A number, string or character
value	An identifier that points to a value that doesn't change
constant	An identifier that points to a value that can change

[3]

2. For each of the following, tick whether they are likely to be a variable name, constant name or value.

	Variable name	Constant name	Value
playerName			
"smith"			
PI			
3.14			

[3]

3. Variable names should be clear and indicate what they will be holding. Which of the following are the best choices for variable names. Tick **three** boxes.

❑ a	❑ p	❑ player	❑ playerName
❑ t	❑ time	❑ timeTaken	❑ tT
❑ s	❑ p_s	❑ playerScore	❑ player score

[3]

10

5. The rules of the language are known as what? Fill in one circle.

❍ Semantics ❍ Syntax
❍ Compilation ❍ Highlighting

[1]

4. Label each part of syntax in the line of code below.

```
playerScore = 10
```

[3]

5. Look at the code on the right.

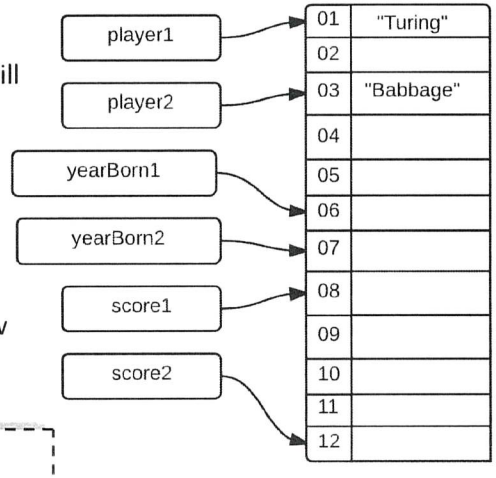
```
player1 = "Turing"
player2 = "Babbage"
yearBorn1 = 1912
yearBorn2 = 1791
score1 = 27
score2 = 31
score1 = score1 + 5
```

a) What type of programming structure is used?

❍ Constants ❍ Selection
❍ Sequence ❍ Equality

[1]

b) Complete the diagram on the right, showing the values that will be stored in memory when the program has finished running. The first two have already been completed.



[4]

c) Complete the line of code below so that score1 is increased by 1.

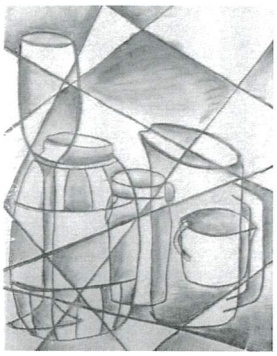
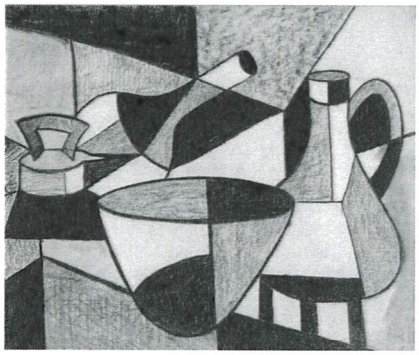
```
score1 = _____ + 1
```

[1]

10

Year 7 Art Homework 3

Name:
Art class:

Create your own abstract still life by adding effects to the outline provided for you. Follow the instructions below. The images above should give you some ideas too.

1) Divide the composition into segments by drawing multiple straight, curved or wiggly lines across the image and / or within the individual shapes
2) Add a range of tonal effects within some of the segments, trying to create a strong contrast across the composition
3) Add straight or curved lines within other segments to create patterns
4) Leave some segments white if you wish