Student Number:

2021 Trial Higher School Certificate Examination

Software Design and Development

General Instructions:

- No reading time
- Working time – 60 minutes
- **Type** the answers in the examination document
  Font: Times New Roman
  Font size: 12
  Font Colour: Black
- No Calculators allowed
- This is an open book examination (only class notes or textbooks are allowed)
- The use of phones and the internet is prohibited; upload your answers to Google Classroom

Total marks: 40

- Attempt ALL questions (A-J)
- You should save your answers as you go

Answer the questions in the spaces provided. At the end of each question there will be guidance provided for the expected length of response, eg, *About 4 lines*

Programming Paradigms

(A)

Consider the following in relation to object-oriented paradigm concepts:

All cars, buses and bicycles are vehicles. All vehicles have travelled a certain distance since new, have a particular number of wheels and seats, use some type of fuel, are able to move and can turn. However, moving and turning cars and buses is quite a different process to moving and turning a bicycle. As a vehicle moves, it travels a certain distance, and the distance travelled since new is stored within the vehicle's odometer. The only way to alter the value in the odometer is to move the vehicle.

Based on the above information:

(i) Describe an example of a class including its data and methods. (2marks, about 3 lines)

(ii) Identify an example of inheritance and use your example to explain how inheritance improves the productivity of programmers. (2marks, about 4 lines)

(iii) Describe an example of polymorphism. (2marks, about 4 lines)

(iv) Identify an example of encapsulation and explain how encapsulation improves the ability to reuse code. (2 marks, about 5 lines)

(B) C#, Java, Python, VB.Net and PHP are all classed as object-oriented programming languages. Describe two factors that influence developers to choose a language. (2 marks, about 5 lines)

(C) In some computer languages, the plus symbol (+) is used to add 2 numbers together and is also used to concatenate (join) two strings together. Describe how this could be considered as an example of polymorphism. (2 marks, about 5 lines)

(D) The following statement briefly describes the differences between two different styles of programming:

"imperative" programming tells the computer how to do something, and as a result what you want to happen will happen, whereas "Declarative" programming tells the computer what you would like to happen, and lets the computer figure out how to do it.

Discuss the differences between these two approaches to programming. In your discussion, give examples of where each style of programming might be used. (3 marks, about 9 lines)

(E) The learning curve required to adopt a new programming language is one factor influencing programmer productivity. Discuss TWO other factors which influence the productivity of programmers and relate these to programming paradigms. (3 marks, about 9 lines)

(F) Francis is developing a program in the logic paradigm to keep track of video games and the platforms they are played on. He has also established a knowledge base on the gaming platforms that he owns.

```
game(COD)
game(Fortnite)
game(Minecraft)
game(GT-Sport)
game(Midtown Madness)
game(Rust)

my_platform(PC)
my_platform(Playstation)

game_platform(X-Box, Midtown Madness)
game_platform(X-Box, Fortnite)
game_platform(X-Box, Minecraft)
game_platform(PC, Minecraft)
game_platform(PC, Rust)
game_platform(PC, Pillars of Eternity)
game_platform(Playstation, Minecraft)
game_platform(Playstation, Fortnite)
game_platform(Playstation, GT-Sport)
```

i_can_play (X) :- game_platform(Y,X), my_platform (Y)

(i) Identify two facts in this code fragment (1 mark, about 1 line)

(ii) Identify the rule in this code fragment and outline the logic behind this rule (1 mark, about 2 lines)

(iii) What would the result of this query be? Explain the process. (2 marks, about 3 lines)

?- i_can_play (“Midtown Madness”)

(iv) Discuss why the logic paradigm is a good approach when developing systems which are designed to represent knowledge. (2 marks, about 5 lines)

(G) A vet uses an object-oriented program to maintain a database of all their customers' pets.

```
Class PET {
    Private –
            Name :     string
            Gender :   character
            RealAge :  integer
    Public –
            Get_Name ( ) :     string {
                  Return Name
             }
            Get_Gender ( ) :   character {
                  Return Gender
             }
             Get_Age ( ) :       integer {
                   Return RealAge
              }
              Birthday ( ) :      void {
                    RealAge = RealAge +1
              }
} // end class PET
```

(i) Using the sample of code given, write a sub-class of PET called DOG that adds an attribute called "Breed" to store the breed of the dog and a method called "Get_Dog_Age" that returns the age of the dog in dog years, where one human year is 7 dog years (e.g. if the RealAge were 3 then the dog age would be 21). An outline of the structure for the code is shown below. You need to re-write this structure including the appropriate public and private code sections.

Sub-Class DOG {

is a PET

Private -

Public –

} // end sub-Class DOG

(2 marks, about 1/4 of the page)

(ii) With reference to the above example. Describe how Polymorphism could be used in defining the method used to calculate the age of different pets, assuming that different animals age at different rates as compared to human years (i.e. 1:7 for dogs, 1:35 for mice, etc.)

(2 marks, about ¼ of the page)

(iii) Describe the use of private and public sections in the definition of a class. (2 marks, about 4 lines)

(H) A scientist is trying to classify various rock materials. Many experiments have been performed to determine the features and characteristics of each sample. Features and characteristics include colour, atomic weight and structure, abundant elements plus location found. With all of this information the scientist has discovered that it is possible to classify any rock material into one of the categories. The scientist wants his assistants to be able to carry out the classifications process without him present and therefore requires a system to be developed which accepts the necessary information about the sample and classifies it.

Recommend the most appropriate paradigm for the solution of this problem. Explain why the other paradigms would be less appropriate. (4 marks, about ½ of the page)

(I) Facts and rules are the basic building blocks of the logical programming paradigm.

The following facts have been defined in Prolog.

feathers (finches)

feathers (parrots)

This means, both finches and parrots have feathers.

Create further Prolog facts and rules based on the following information:

Facts:

- Bats, parrots and finches have wings (1/2 mark)
- Echidnas, parrots and finches lay eggs (1/2 mark)
- Bats and cows give birth to live young (1/2 mark)
- Cows, bats and echidnas suckle their young (1/2 mark)

Rules:

- All birds lay eggs and have feathers and wings (1mark)
- All mammals give birth to live young and suckle their young (1mark)

(you must include the right syntax for the rules, write your answers below:)

Facts are:

Rules are:

(J) Explain ‘backward chaining’ in the logic paradigm and provide a real-life example where it can be used. (2 marks, about 4 lines)

End of Trial exam, check your answers and upload to Google classroom

(A) 8 marks

(i)2 marks

A class could be created called Vehicle which would include the data (or attributes) Wheels, seats, fuel and odometer. Methods(or operations) would include turn and move.

(ii) 2 marks

Three sub-class (or child classes), say Car, Bus and Bicycle could inherit data and methods from the parent class Vehicle. This reduces the need to write the sub-classes from scratch as the code within the Vehicle class is reproduced within the sub-classes. In addition, once the code has been thoroughly tested it can be relied upon each time it is reused. The Bicycle class would have its own distinct Move and Turn methods rather than inheriting these from the parent Vehicle class.

(iii) 2 marks

Assume one object from each of the classes Car, Bus and Bicycle has been created. The methods Move and Turn can be used to manipulate all three objects. However the Move method for the Bicycle object performs quite different processing to the Move method of the Car and Bus objects. Similarly for the Turn method. Both the Move and Turn methods are said to be polymorphic.

(iv) 2 marks

Encapsulation means the value of data (attributes) of an object can only be altered by that object's methods. In the example the Odometer attribute can only be altered by the Move method of that object. Encapsulation allows programmers to create robust objects that are suitable for reuse. As data can only be changed by methods the programmer can test and validate all inputs thoroughly within the objects methods before any of the object's data is changed. This means unexpected inputs can always be dealt with gracefully by the object rather than causing errors to occur within the larger program.

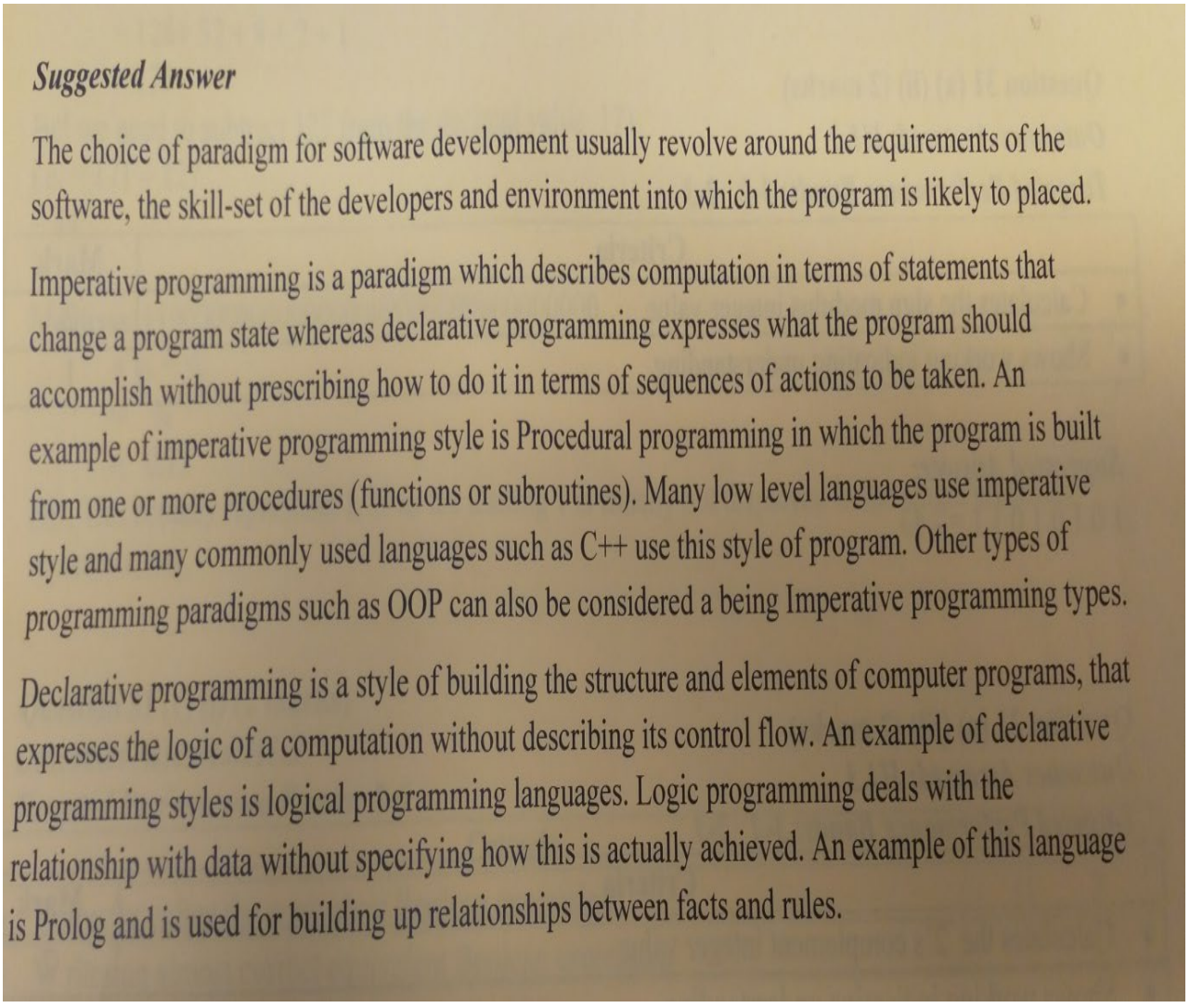
(B) 2 marks

Programmer generally use a language that is best suited to solve the problem. However, they also tend to stick with languages they are most familiar with. Some languages are not designed to resolve particular problems, for example you would not use an imperative language to query a database. Imperative and OOP languages are not suited to solving logic based problems. Another reason for choosing a specific language is the level of support and training available and the po9lularity of the language. E.g. Python and C#.

(C) 3 MARKS

POLYMORPHISM REFERS TO THE ABILITY OF THE SAME COMMAND TO PROCESS OBJECTS FROMT HE SAME SUPER CLASS DIFFERENTLY depending on their subclass. This means that the language interpreter or compiler can determine at run time which method to apply based on the date type of the parameter. In the case of the plus (+) symbol, it performs different functions depending on the data type passed to it. If two numeric data types are passed to the plus method, the numbers are added. If two string data types are passed to the method, the strings are concatenated. Same method, different outcomes depending on the types of data passed to it.

(d) 3 marks



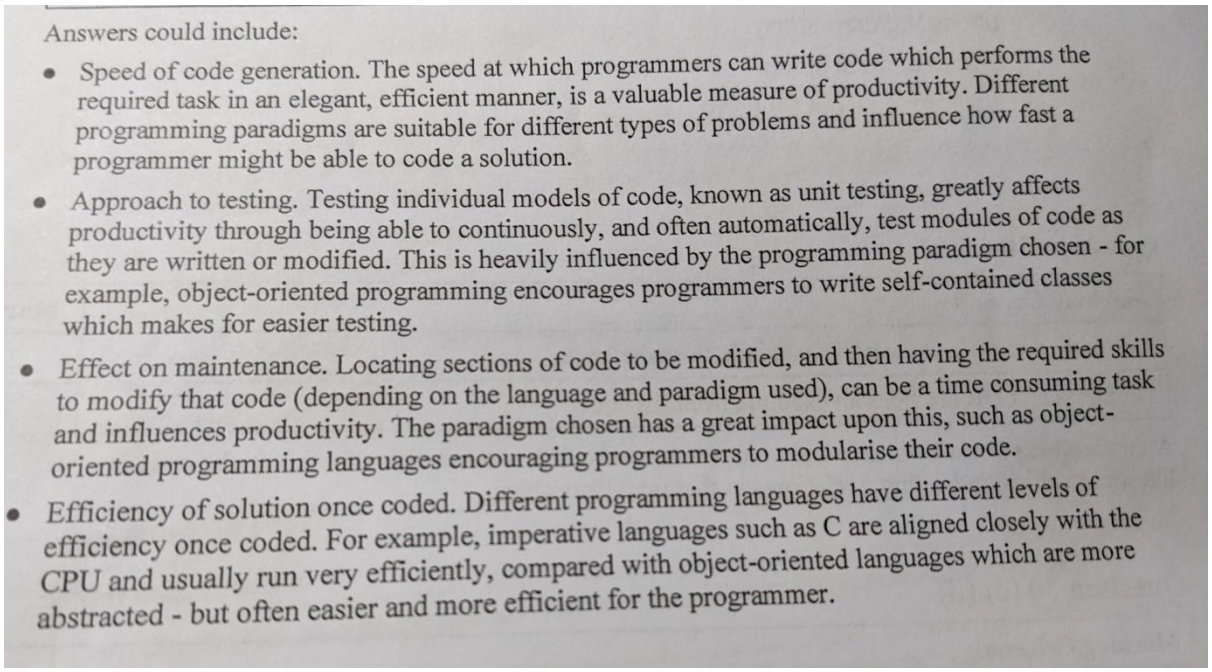
Suggested Answer

The choice of paradigm for software development usually revolve around the requirements of the software, the skill-set of the developers and environment into which the program is likely to placed.

Imperative programming is a paradigm which describes computation in terms of statements that change a program state whereas declarative programming expresses what the program should accomplish without prescribing how to do it in terms of sequences of actions to be taken. An example of imperative programming style is Procedural programming in which the program is built from one or more procedures (functions or subroutines). Many low level languages use imperative style and many commonly used languages such as C++ use this style of program. Other types of programming paradigms such as OOP can also be considered a being Imperative programming types.

Declarative programming is a style of building the structure and elements of computer programs, that expresses the logic of a computation without describing its control flow. An example of declarative programming styles is logical programming languages. Logic programming deals with the relationship with data without specifying how this is actually achieved. An example of this language is Prolog and is used for building up relationships between facts and rules.

(E) 4 marks



Answers could include:

- Speed of code generation. The speed at which programmers can write code which performs the required task in an elegant, efficient manner, is a valuable measure of productivity. Different programming paradigms are suitable for different types of problems and influence how fast a programmer might be able to code a solution.
- Approach to testing. Testing individual models of code, known as unit testing, greatly affects productivity through being able to continuously, and often automatically, test modules of code as they are written or modified. This is heavily influenced by the programming paradigm chosen - for example, object-oriented programming encourages programmers to write self-contained classes which makes for easier testing.
- Effect on maintenance. Locating sections of code to be modified, and then having the required skills to modify that code (depending on the language and paradigm used), can be a time consuming task and influences productivity. The paradigm chosen has a great impact upon this, such as object-oriented programming languages encouraging programmers to modularise their code.
- Efficiency of solution once coded. Different programming languages have different levels of efficiency once coded. For example, imperative languages such as C are aligned closely with the CPU and usually run very efficiently, compared with object-oriented languages which are more abstracted - but often easier and more efficient for the programmer.

(F)

(i) 1 marks

game(C0D)

game(Fortnite)

(ii) 2 marks

The rule: i_can_play(X) :- game_platform (Y,X), my_platform(Y) establishes what game(X) the user is capable of playing based on the platform of the particular game (Y,X) and the platform the user has (Y).

(iii) 2 marks

The query would be applied against the “i_can_play” rule. The rule is checked against the facts and would return FALSE due to “Midtown Madness” only being available on X-Box which is a platform the user does not have.

(iv) 2 marks

The logic paradigm of programming is based upon reasoning and logical thought which reflects how humans think, reason, and make decision. As the logic paradigm relies on the definition of facts which are known to be true this tends well towards the representation of knowledge which is essentially the same thing. Then the focus of the developer can be on how to accurately store and represent that info or knowledge-and the programming language, using the logic paradigm, can manage much of the process of taking a problem, applying it towards the facts and returning a solution.

(G)

(i) 2 marks

```
Solution 1: Sub-Class DOG   {
                            is a PET
                            Private –
                                Breed : string
                            Public –
                                Get_Dog_Age ( ) : integer {
                                        Return (RealAge * 7)
                            }
                        } // end sub-Class DOG
Solution 2 : Sub-Class DOG  {
                            is a PET
                            Private –
                               Breed : string
                            Public –
                              Get_Breed ( ) : string {
                                    Return Breed
                              }
                              Get_Dog_Age( ) : integer {
```

```
                    Real_age : integer
                    Dog_Age = RealAge * 7
                    Retrun Dog_Age
                }
        } // end sub-Class DOG
```

(ii) 2 marks

Instead of defining a new method in each class called “Get_Dog_Age”, “Get_Mouse_Age” “Get_Cat_Age”, etc, each sub-class could simply redefine the “Get_Age” method to include the relevant age calculation. The overriding of the original method by a new method in the sub-class with the same name as the method in the parent class is called Polymorphism.

In a sub-class MOUSE

```
Sub-Class MOUSE    {
        is a PET
        Private –
          …….
          …….
        Public –
            Get_Age ( ) : integer {
                    Return (RealAge * 35)
        }
} // end sub-Class MOUSE
```

In a sub-class DOG

```
Sub-Class DOG      {
        is a PET
```

```
        Private –
          …….
        ……..

        Public –
            Get_Age ( ) : integer {
                    Return (RealAge * 7)
        }
} // end sub-Class DOG
```

(iii) 2 marks

The private section is encapsulated or hidden from other classes or code. This is often used to contain the variable attributes of a class. On the other hand, the public section is accessible outside of the class and is often used to contain member functions that provide access to the private attributes. (Methods in the public section are able to access private sections within the same class)

(H) 4 marks, logic, student response

(I)

4 marks in total

facts ½ mark each and rules one mark each

Facts:

Wings (bat)

Wings(parrot)

Wings(finch)

LayEggs(echidna)

LayEggs(parrot)

LayEggs(finch)

LiveYoung (bat)

LiveYoung(cow)

SuckleYoung (cow)

SuckleYoung (bat)

SuckleYoung (echidna)

Rules:

Birds (X) :- LayEggs(X), Feathers (X), Wings(X)

Mammals (X) :- LiveYoung(X), SuckleYoung (X)

(J) 2 marks

Backward chaining is the process of arriving at a conclusion from a stated set of conditions. It assumes that a particular solution is true and then asks questions to verify that the necessary conditions are present. One example could be medical diagnosis. A patient may be suspected of having a particular illness. The program searches for all rules in relation to the illness and asks questions regarding those rules. If all conditions are met, then the initial diagnosis is correct.