|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Problem Statement:  These characteristics include:   * **large volumes** of data all requiring similar processing * requirement for **information to be available** from several different locations * requirement for very **accurate calculations** * **duplicated effort** is involved * manual methods **are too slow** * data needs to be **constantly up-to-date and accessible** * **Validation** and **verification** | | | | | [6] |
|  | The system developer has drawn up a list of activities and their likely duration   |  |  |  | | --- | --- | --- | | **Activity** | Description | Weeks to complete | | A | Write requirement specification | 1 | | B | Produce program design | 1 | | C | Write module code | 7 | | D | Module testing | 2 | | E | Integration testing | 2 | | F | Alpha testing | 2 | | G | Install software and carry out acceptance testing | 2 | | H | Research and order hardware | 1 | | J | Install delivered hardware | 3 | | K | Write technical documentation | 4 | | L | Write user training guide | 2 | | M | Train users on installed hardware and software | 1 | | N | Sign off final system | 1 | | | | | |  |
|  |  | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Activity** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | A |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | B |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | C |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | F |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | G |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | H |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | J |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | K |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | L |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | M |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | N |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | Week number | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | | | [2] |
|  |  | | | week number 18 | | [1] |
|  |  | | | |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **Activity** |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | A |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | B |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | C |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | D |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | E |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | F |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | G |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | H |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | J |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | K |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | L |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | M |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | N |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | Week number | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | | | [9] |
|  |  | | | **week number: 26** | | [1] |
|  |  | | | **Team work is actually the collective effort of each and every team member to achieve their assigned goal.**  **No member can afford to sit back and expect the other member to perform on his behalf. The team members must be committed towards their team as well as their organization to avoid conflicts. Nothing productive comes out of unnecessary conflict and in turn diverts the concentration and focus of the team members. Every member should adopt an adjusting and a flexible attitude. One should consider his team members as a part of his extended family all working together towards a common goal. The team members must be dependent on each other for the best to come out.** | | [2] |
|  | The use of verification and validation checks, firewalls, anti-virus software, passwords and encryptions to protect the computer system over the network.  **Encryption**   * Data can be encrypted so that they appear garbled unless you know how to decrypt the data. * This can prevent unauthorised access as the data will be meaningless.   **Passwords**   * Passwords can be applied to computer systems so users cannot access the data unless they know the password. * Passwords can be applied to individual files. * A hierarchy of passwords may exist. * Passwords should be changed regularly in case they become generally known.   **Virus guards**   * A virus is a small program that attaches itself to your files. * When you use a file with a virus attached it loads itself into the memory of the computer. * It is able to replicate itself so that every time a new file is used it attaches itself to that one as well. | | | | | [3] |
|  | **Capturing Visitor Particulars**   |  |  |  | | --- | --- | --- | | Salutation | **Mr** | **⭠Dropdown list** | | Name | **John Tan** | **⭠Text box** | | Address | **Blk 107 Telok Blangar Avenue 5 #07-108** | **⭠Text box** | | Gender | **🞊 F 🞊 M** | **⭠Radio button** | | Mobile number | **99912345** |  | |  | **Save** |  | | | | | |  |
|  |  | **Repeatedly**  **breaking a problem down**  **into simpler problems**  **until the problems can be solved easily** | | | | [3] |
|  |  | **Modules can be kept in a library**  **and re-used in other solutions**  **Many programmers can work on same problem**  **as each can be given different modules to solve**  **Easier to debug**  **as modules are small**  **Easier to maintain and modify**  **as modules can be removed/added easily** | | | | [3] |
|  |  | **Include comments**  **which explain what the code is doing**  **Use indentation**  **to show blocks of code**  **Use meaningful variable names**  **so that others know what they represent** | | | | [3] |
|  |  | **first = 1, last = 142**  **mid = 71, last = 70 1 mark**  **mid = 35, first = 36 1 mark**  **mid = 53, last = 52 1 mark**  **mid = 44, last = 43 1 mark**  **JOSEPH found** | | | | [4] |
|  |  | **Thus 8 elements at most must be examined**  **Because each split halves number of elements left**  **Thus the elements examined are 128, 64, 32, 16, 8, 4, 2 and 1** | | | | [3] |
|  |  | | |  |  | | --- | --- | | Class name: Visitor | | | attributes | Passport\_number | |  | Claim\_amount | |  | Type\_of\_claim | | Methods |  | |  | Get\_Passport\_number | |  | Set\_Passport\_number | |  | : | |  | Display |  |  |  | | --- | --- | | Class name: Visitor\_GST\_by\_HP | | | attributes | rebate | | Methods | Get\_rebate | |  | Set\_rebate | |  | Display |  |  |  | | --- | --- | | Class name: Visitor\_GST\_by\_Cash | | | attributes | service\_charge | |  | exchange\_rate | | Methods |  | |  | Get\_service\_charge | |  | Set\_service\_charge | |  | : | |  | Display | | | | [6] |
|  |  | |  | | |  |
|  |  | |  | | Encapsulation and how classes support information hiding and implementation independence. | [3] |
|  |  | | (i) | | **Encapsulation is combining together methods and attributes as a single object type.**  **e.g. Visitor is a class which combining together attributes and methods as a single object type** |  |
|  |  | |  | | Inheritance and how it promotes software reuse. | [2] |
|  |  | | (ii) | | **Inheritance means that classes can inherits the attribute and methods from parent class**  **e.g. Visitor\_GST\_by\_Cash and Visitor\_GST\_by\_HP are classes inherited together attributes and methods from parent class Visitor.** |  |
|  |  | |  | | Polymorphism and how it enables code generalisation. | [2] |
|  |  | | (iii) | | **Polymorphism means that inherited methods don't have to behave in exactly the same way**  **e.g. The Display method of Visitor\_GST\_by\_Cash don’t have to behave in exactly as Display method of parent class Visitor…………….** |  |
|  |  | |  | | What is computational thinking and why it is important? | [5] |
|  |  | | (iv) | | **Computational thinking requires abstraction of both data and procedures and that computational thinking is the means to practical and effective solutions for real-world problems.**  **Abstraction - It means taking away the unneeded information to leave just what is needed to solve the problem.**  **It is part of computational thinking which allows us to look at a problem in ways which are easier to deal with.** |  |
|  |  | | **Hides data – can only access data via methods**  **Reuse is easier**  **Methods can be changed without affecting how they are used.** | | | [2] |
| 8. |  | **Cloud computing consists of 3 layers in the hierarchy and these are as follows:  1. Infrastructure as a Service (IaaS) provides cloud infrastructure in terms of hardware like memory, processor speed etc. …. 2. Platform as a Service (PaaS) provides cloud application platform for the developers. …. 3. Software as a Service (SaaS) provides cloud applications which are used by the user directly without installing anything on the system. …** | | | | [3] |
|  | (b) | **Drawbacks**  **Loss of control of the data**   * **as the data is controlled by a data storage provider** * **the user needs to be able to trust the service provider and its staff** * **security issues with the storing of the data with the provider** * **fire/flood/electricity spikes can affect data without the user knowing**   **If data is not encrypted**   * **can lead to security issues due to loss of control/hacking**   **If connection is lost, then the cloud cannot be accessed**   * **data could be lost due to this**   **If there is a mass/lot of data, the data could be split over several servers**   * **may have problems retrieving the data** * **may lead to security issues** * **data may be lost during the saving process**   **Cost of storing data may be higher than physical methods**   * **most computers have storage so don’t need to purchase one** * **private clouds can be expensive**   **Difficult to delete all copies of a file**   * **as the data is stored/backed up several times**   **Benefits**  **The data is accessible anywhere**   * **helps to migrate data from one device to another** * **helps to synchronise data**   **Don’t have to buy storage devices**   * **some devices do not have hard discs/storage** * **cheaper than buying storage devices** * **the cloud can be used for all storage requirements**   **More storage space than using physical devices**   * **the cloud has lots of storage**   **Don’t have to manage the storage**   * **the service provider manages the storage** * **user does not have to worry about security measures as service provider secures data**   **Allows many users to access the data**   * **as the data is stored centrally**   **Only pay for the storage you use**   * **much of the space on the hard disc may be wasted**   **The data is regularly backed up** | | | | [4] |
| 9. | (a) | **Space**  **SpaceList**  **Rental**  **Tenant**  Tenant (**tenantID**, companyName, director, contactNo, companyAddress, retailType)  Rental (**tenantID**, **spaceID**, **rentalDateStart**, rentalDateEnd)  SpaceList (**spaceID**, level, unitno, spaceCategory)  Space (**spaceCategory**, areaSize, rentalRate) | | | |  |
|  | (b) | * The data will be independent of the application. * The DBMS will maintain data consistency. * Data integrity will be improved. * The data will be more available. * The data can be centrally controlled. * It is easier to keep the data secure. | | | |  |
| 10. | (a) | Papaya  Mango  Strawberry  Mango  Strawberry  Mango  Strawberry | | | |  |
|  | (b) | **RootPtr** = 1   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **Index** | **LeftPtr** |  | **Fruit** |  | **RightPtr** | | 1 | 2 |  | Papaya |  | 3 | | 2 | 4 |  | Mango |  | 5 | | 3 | 6 |  | Strawberry |  | 7 | | 4 | 0 |  | Durian |  | 0 | | 5 | 0 |  | Orange |  | 0 | | 6 | 0 |  | Rambutan |  | 0 | | 7 | 0 |  | Watermelon |  | 0 | | ... … |  |  |  |  |  | | ... … |  |  |  |  |  | | 1000 |  |  |  |  |  | | | | |  |
|  | (c) | REPEAT  IF Fruit[Current] = SearchFruit THEN **[1]**  OUTPUT "Found" **[1]**  IsFound ← TRUE **[1]**  ELSE  IF SearchFruit > Fruit[Current] THEN **[1]**  Current ← RightPtr[Current] **[1]**  ELSE  Current ← LeftPtr[Current] **[1]**  ENDIF  ENDIF  UNTIL Current = 0 OR IsFound = TRUE | | | |  |
| 11. | (a) | Character ‘4’ is ASCII 52 (decimal)  Therefore: 52 (den) -> 0011 0100 (8-bit binary) | | | |  |
|  | (b) | ‘W’: 0101 0111 | | | |  |
|  | (c) | Adv: Use less bits to store characters, save space  Disadv: Cannot represent other languages other than those using alphabets | | | |  |
| 12. |  | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **CONDITIONS** | 1 | 2 | 3 | 4 | 5 | | Purchases totalling > $30 | Y | Y | Y | Y | N | | Purchases totalling > $200 | Y | Y | N | N | - | | Have member card | Y | N | Y | N | - | | **ACTIONS** |  | | | | | | No discount |  |  |  | X | X | | Discount of 5% |  | X | X |  |  | | Discount of 10% | X |  |  |  |  | | | | |  |
| 13. | (a) | 98, 12, 23, 8, 74, 30, 62  12, 23, 8, 74, 30, 62, 98  12, 8, 23, 30, 62, 74, 98  8, 12, 23, 30, 62, 74, 98  8, 12, 23, 30, 62, 74, 98  8, 12, 23, 30, 62, 74, 98  8, 12, 23, 30, 62, 74, 98  8, 12, 23, 30, 62, 74, 98 | | | |  |
|  | (b) | 98, 12, 23, 8, 74, 30, 62  12, 98, 23, 8, 74, 30, 62  12, 23, 98, 8, 74, 30, 62  8, 12, 23, 98, 74, 30, 62  8, 12, 23, 74, 98, 30, 62  8, 12, 23, 30, 74, 98, 62  8, 12, 23, 30, 62, 74, 98 | | | |  |
|  | (c) | Bubble sort: for each pass, each element will always be compared to the other unsorted elements.  Insertion sort: for each pass, each element will be compared until it is swapped into an already sorted elements into its correct position.  Hence relatively fewer comparisons and Insertion sort is faster than Bubble sort. | | | |  |