|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Question** | | | **Answer** | **Mark** |
| 1 | a |  | 1 mark per bullet point   * Finding a random number between 1 and 4 * Checking the random number using an IF statement * Assigning a special ability   e.g  rnum = randomNum(1,4)  IF rnum == 1 THEN  special\_ability = “fire”  ELIF rnum == 2 THEN  special\_ability = “water”  ELIF rnum == 3 THEN  special\_ability = “earth” | 3 |
| 1 | b |  | 1 mark per bullet point.   * Asking how many levels * total =0, max = 0 * min = 1000 or something bigger than 1000 * Loop that goes for the number of levels * Ending the loop correctly * Asking for coins * Calculating total * Calculating max * Calculating min * Display total, min, max   2 marks for efficiency  e.g  levels = INPUT (“How many levels have you reached?”)  total = 0  min = 1000  max = 0  FOR i=0 TO levels  coins = INPUT (“How many coins did you collected in level”+i)  total = total + coins  IF coins > max THEN  max = coins  ENDIF  IF coins < min THEN  min = coins  ENDIF NEXT i  OUTPUT (“The total number of coins you have so far is”,total)  OUTPUT (“The highest number of coins you have so far is”,max)  OUTPUT (“The least number of coins you have so far is”,min) | 12 |
| 2 | a |  | 1 bit | 1 |
| 2 | b |  | 1 mark for each line that is correct  0 1 1 1 0 1 0 0 0 1  1 1 1 1 1  1 0 0 0 1  0 1 1 1 0 | 5 |
| 2 | c | i | Metadata is information about the image that are necessary for the image. | 1 |
| 2 | c | ii | 1 mark per bullet point upto 2 marks:   * Height of the image * Width of the image * Number of bits per pixel | 2 |
| 3 | a |  | 1 mark per bullet point:   * Normal Data * Erroneous / Invalid Data * Boundary Data | 3 |
| 3 | b |  | 1 mark for each missing box. Similar answers for column 3 (Reason of test) may be accepted as long as they specify the correct reason of testing. | 6 |
| 4 | a |  | 1 mark for each bullet point   * Show Lower, Middle and Upper Bound. * Checking that the Middle point (112) is not 140 * Choosing the side above the middle point [140,160,162,180] * Checking that the Middle point (160/162 both accepted) is not 140 * Choosing the side below the middle point [140] / [140,160] * Finding 140 as the middle point | 6 |
| 4 | b |  | When the list is not sorted. | 1 |
| 5 |  |  | 1 mark for each point   * Showing the list being splitted into sorted and unsorted lists * Putting 21 to the correct position * Putting 10 to the correct position * Putting 45 to the correct position * Putting 15 to the correct position * Showing that 15 changes positions with 45 first and then 21 | 6 |
| 6 | a |  | Compression is used to reduce the size of a file [1] in order to free up some space [2] |  |
| 6 | b |  | Lossy  Lossless | 2 |
| 6 | c |  | Lossy compression [1]  1 mark for each bullet point upto 1 mark,   * Because it reduces the size of the file more than lossless [1] * Because the data that will be lost with lossy are sounds that cannot be heard by humans [1] | 2 |
| 7 | a | i | 00110110 | 1 |
| 7 | a | ii | 10101000 | 1 |
| 7 | b | i | 52 | 1 |
| 7 | b | ii | 138 | 1 |
| 7 | c |  | 01010100 | 1 |
| 7 | d |  | 00001101 | 1 |
| 7 | e |  | 1011 1011  1011 0111 +  -----------------  101110010 | 2 |
| 7 | f |  | Binary Overflow | 1 |
| 7 | g |  | Computers have special registers to note overflows [1]  Certain bits known as flags will be set to remind the computer that a certain overflow has occurred [1] | 2 |
| 8 | a |  | 1 mark for each bullet point, upto 2 marks   * Low-level languages are quicker to be translated to binary hence faster to be used by the CPU * Code can be executed faster * Used to write programs for specific hardwares | 2 |
| 8 | b |  | 1 mark for naming each translator and 1 mark for each explanation   * Compilers   + Convert all the code in one go and provide an error log at the end * Interpreters   + Convert one line at a time and stop when they find an error | 4 |
| 9 | a |  | X = (D OR (B AND C)) OR (A OR B) | 3 |
| 8 | b |  |  | 8 |