Programming is often a creative problem solving activity and there can be many/different ways to tackle a task. Candidates who show alternative and correct approaches will also be awarded equivalent credit.

|  |
| --- |
| **Task 1.1** |
| **Evidence 1 [9]** |
| programming style - meaningful identifier names, appropriate white space and comments |
| open file in correct mode and close file with exception handling |
| skip header line |
| variables and data structure initialisation |
| read data into appropriate data structure |
| appropriate scoring system to determine rank |
| loop and update scores |
| sort in descending scores order to determine and update rank |
| for teams with same score, sort in ascending team name order |
| display correct rank and team name and medal string |
| **Evidence 2 [1]** |
| screenshot of correct ranked top 3 teams with tied teams |
| **Task 1.2** |
| **Evidence 3 [4]** |
| loop to check for terminating condition and prompt for team name input |
| linear search by team name |
| extract and display name and medal / participation |
| output end message |
| **Evidence 4 [1]** |
| screenshot of one or more team results with terminating condition met |
| **Task 2.1** |
| **Evidence 5 [3]** |
| open file in correct mode and close file with exception handling and read into appropriate variable / data structure |
| appropriate string processing to extract last data item |
| output last data item |
| **Evidence 6 [1]** |
| screenshot of last data item |
| **Task 2.2** |
| **Evidence 7 [5]** |
| appropriate mapping of hexadecimal to binary |
| appropriate mapping of binary to octal |
| convert hexadecimal to binary |
| regroup from four binary to three binary digits |
| convert binary to octal |
| **Evidence 8 [1]** |
| screenshot of correct octal value |
| **Task 2.3** |
| **Evidence 9 [3]** |
| normal data (1 to E) |
| boundary data (0 or F) |
| erroneous data (G) |
| **Evidence 10 [2]** |
| appropriate and specific error message for erroneous data |
| correct handling of normal / boundary data |
| **Task 3.1** |
| **Evidence 11 [6]** |
| open files in correct modes and close files with exception handling [2] |
| proper definition and initialisation of variables and dictionary |
| read data from file and extract required data fields |
| insert as dictionary entries (key-value pairs) |
| correct loop condition |
| loop to output first 10 dictionary entries (key-value pairs) |
| **Evidence 12 [1]** |
| screenshot of first 10 dictionary entries (key-value pairs) |
| **Task 3.2** |
| **Evidence 13 [16]** |
| good organisation of OOP and main program structure showing modularisation and decomposition |
| proper linked list node class initialisation |
| proper linked list class initialisation |
| for schools A-D |
| appropriate insert and update/delete methods for linked list class [4] |
| determination of achievement level [2] |
| determination of psle score [2] |
| preprocessing of students dictionary (eg using appropriate sorting algorithm) [3] |
| allocate to appropriate schools A-C linked lists [3] |
| allocate to school D linked list [2] |
| display linked list contents [2] |
| **Evidence 14 [4]** |
| screenshot for first 5 school A allocation |
| screenshot for first 5 school B allocation |
| screenshot for first 5 school C allocation |
| screenshot for first 5 school D allocation |
| **Task 3.3** |
| **Evidence 15 [5]** |
| appropriate search for student |
| appropriate updating of citizenship status |
| appropriate updating of secondary 1 posting order [2] |
| display updated allocations [2] |
| **Evidence 16 [2]** |
| screenshot showing updated posted school order for student P351 |
| screenshot showing updated posted school order for student P365 |
| **Task 3.4** |
| **Evidence 17 [5]** |
| appropriate search for student |
| appropriate updating of citizenship for student withdrawal |
| appropriate removal of allocation |
| appropriate replacement allocation [2] |
| display updated allocation [2] |
| **Evidence 18 [1]** |
| screenshot showing updated secondary 1 for student P286 |
| **Task 4.1** |
| **Evidence 19 [13]** |
| programming style - meaningful identifier names, appropriate white space and comments |
| open files in correct modes and close files with exception handling [2] |
| proper definition and initialisation of variables and data structure(s) |
| read data from file and store in appropriate data structure(s) |
| nested loop with correct condition/bounds [2] |
| correct intermediate processing logic [2] |
| proper assignment of A, B, C, D [2] |
| compute D-B-C+A |
| write to IMAGE1.OUT |
| output D-B-C+A |
| **Evidence 20 [2]** |
| screenshot of correct integral image |
| correct computed value of D-B-C+A |
| **Task 4.2** |
| **Evidence 21 [13]** |
| open files in correct modes and closes file with exception handling [2] |
| proper definition and initialisation of variables and data structure(s) |
| pseudo-random number generation of IMAGE2.IN values [2] |
| write to and read from IMAGE2.IN |
| read data from file and store in appropriate data structure(s) |
| proper generalisation of magic() generalisation (m, n >= 8) [3] |
| nested loop with correct condition/bounds [2] |
| correct intermediate processing logic [2] |
| proper assignment of updated A, B, C, D [2] |
| compute and output updated D-B-C+A |
| write to IMAGE2.OUT |
| **Evidence 22 [2]** |
| screenshot of correct integral image |
| correct computed value of D-B-C+A |

## Common Issues

General:

* Please do not miss out your identification (statutory name, centre number (3042 not Dunman High School), index number (4-digit)) and programming language version (Python 3.6)
* Please use a fixed width font (eg Courier New and Consolas) instead of relying on the default (usually Times New Roman or Arial) in the provided EVIDENCE.DOCX for better alignment and hence readability of program code. An appropriate font size is 10 and you can/should ensure minimal wraparound for readability.
* There is generally lack of program comments, or comments are often an afterthought. While the best code is simple, clean/r and self-documenting, there is alway room and credit to insert appropriate comments (even as an afterthought) for A-level practical exam.
* There is not sufficient evidence to show that candidates spend some of their time in maximising evidence and thus credit for EVIDENCE.DOCX. Partial credit accumulates and can end up to be significant.

Task specific:

Task 1

* Key step is to devise a scoring system to determine rank order. Suitable ones include
* 1.1 is tedious but 1.2 is trivial/simple since the data is already sorted by global score (hence medal) order. For this task, both parts are also independent. Go for easy kill to maximise score.

Task 2

* Simple string manipulating/processing task. Key is to detect pattern.
* Key step to 2.2 is to use mappings and regrouping of binary digits. You can always quickly verify the anwser with either built-in Python functions or Windows Calculator.
* Key step to 2.3 is to select appropriate categories of normal, boundary and erroneous data

Task 3

* Implement and adapt linked list class initialisation and methods
* It is acceptable to do preprocessing involving array/list and/or dictionary before batch inserting/updating to linked lists, though you would also need to implement the appropriate linked list methods for ad hoc transactions
* Not much evidence for the later tasks for more comments :)

Task 4

* Key step is to do systemic row/column then column/row processing to computing accumulated sum. It applies to all problems that when we break down a complex problem into simpler manageable subproblems then it is easier to tackle these subproblems and isolate troubleshooting.
* Not much evidence for the later tasks for more comments :)