**Worksheet 6A: Testing (Part 1)**

Thi Van Anh DUONG Student ID: 90023112

Diploma of Information Technology, Curtin College

ISEN1000 : Introduction to Software Engineering

Coordinator: Khurram Hameed

11 April 2022

## **Student Declaration of Originality**

|  |  |
| --- | --- |
|  | This assignment is my own original work, and no part has been copied from another student’s work or source, except where it is clearly attributed. |
|  | All facts/ideas/claims are from academic sources are paraphrased and cited/referenced correctly. |
|  | I have not previously submitted this work in any form for THIS or for any other unit; or published it elsewhere before. |
|  | No part of this work has been written for me by another person. |
|  | I recognise that should this declaration be found to be false, disciplinary action could be taken and penalties imposed in accordance with Curtin College policy. |

**Electronic Submission:**

|  |  |
| --- | --- |
|  | I accept that it is my responsibility to check that the submitted file is the correct file, is readable and has not been corrupted. |
|  | I acknowledge that a submitted file that is unable to be read cannot be marked and will be treated as a non-submission. |
|  | I hold a copy of this work if the original is damaged, and will retain a copy of the Turnitin receipt as evidence of submission. |

1. **Equivalence Categories**
2. Submodule calcGrade

Imports: mark (integer)

Exports: grade (string)

Calculates a grade, given a mark. For marks less than 50, the grade is "F". For marks from 50 to 59, the grade is "5". For 60 to 69, the grade is "6", and so on up to "10". If mark is invalid, calcGrade will export the empty string "".

|  |  |  |
| --- | --- | --- |
| **Category** | **Test Data** | **Actual Output** |
| < 0 |  | “ ” (empty string) |
| > 0 to < 50 |  | “F” |
| <= 50 to <= 59 |  | “5” |
| <= 60 to <= 69 |  | “6” |
| <= 70 to <= 79 |  | “7” |
| <= 80 to <= 89 |  | “8” |
| <= 90 to <= 99 |  | “9” |
| = 100 |  | “10” |
| >100 |  | “ ” (empty string) |

|  |  |  |
| --- | --- | --- |
| **Category** | **Test Data** | **Actual Output** |
| < 0 | Mark = -9 | “ ” (empty string) |
| < 50 | Mark = 32 | “F” |
| <= 50 to <= 59 | Mark = 55 | “5” |
| <= 60 to <= 69 | Mark = 63 | “6” |
| <= 70 to <= 79 | Mark = 72 | “7” |
| <= 80 to <= 89 | Mark = 88 | “8” |
| <= 90 to <= 99 | Mark = 91 | “9” |
| = 100 | Mark = 100 | “10” |
| >100 | Mark = 102 | “ ” (empty string) |

1. Submodule roomVolume

Imports: width, length, height (reals)

Exports: volume (real)

Calculates the volume of a room, but only if the imported width, length and height are valid. To be valid, width must be at least 2 (metres), length 2.5, and height 3. For invalid imports, this submodule will return 0.

Note: Width (W)

Height (H)

Length (L)

And the values are represented here are in meters

|  |  |  |
| --- | --- | --- |
| **Category** | **Test Data** | **Actual Output** |
| W >= 2, H >= 3, L >= 2.5 | W = 3,H = 7.1 , L = 4.5 | 15 |
| W < 2, H >= 3, L >= 2.5 | W = 1.5,H = 6.7 , L = 9.7 | 0 |
| W >= 2, H < 3, L >= 2.5 | W = 2,H = 2.9 , L = 4.5 | 0 |
| W >= 2, H >= 3, L < 2.5 | W = 4,H = 5.5, L = 2 | 0 |
| W < 2, H < 3, L >= 2.5 | W = 1.9,H = 2.1 , L = 7.6 | 0 |
| W < 2, H >= 3, L < 2.5 | W = 0.2,H = 3.2, L = 1.2 | 0 |
| W >= 2, H < 3, L < 2.5 | W = 8.2,H = 2.6, L = 0.7 | 0 |
| W < 2, H < 3, L < 2.5 | W = 1.3,H = 1.7, L = 1.8 | 0 |

1. Submodule charCase

Imports: checkUpper (boolean), ch (character)

Exports: isCase (boolean)

Checks whether or not ch is an upper/lowercase letter. If checkUpper is true, the submodule checks whether ch is uppercase, and return true/false accordingly. If checkUpper is false, the submodule instead checks whether ch is lowercase.

Note: 1 is TRUE and 0 is FALSE

Check uppercase = 1

|  |  |  |
| --- | --- | --- |
| **Category** | **Test Data** | **Actual Output** |
| >= A to <= Z | F | 1 |
| >= 0 to <= 9 | 8 | 0 |
| >=! to <= \_ | & | 0 |
| = “ ” | “ ” | 0 |

|  |  |  |
| --- | --- | --- |
| **Category** | **Test Data** | **Actual Output** |
| >= a to <= z | d | 1 |
| >= 0 to <= 9 | 2 | 0 |
| >=! to <= \_ | # | 0 |
| = “ ” | “ ” | 0 |

Combine:

|  |  |  |
| --- | --- | --- |
| **Category** | **Test Data** | **Actual Output** |
| >= A to <= Z | S | 1 |
| >= a to <= z | r | 1 |
| >= 0 to <= 9 | 7 | 0 |
| = “ ” | “ ” | 0 |
| >=! to <= \_ | % | 0 |

(d)

Submodule substr

Imports: str1, str2 (strings)

Exports: s (string)

Determines whether one string occurs inside the other. If it does, the submodule returns whichever string is shorter. If not, it returns the the empty string "". Note that the empty string is, by definition, contained inside every string (including itself).

For instance, if str1 is "conscience" and str2 is "science", then this submodule returns "science". If both imported strings are "xyz", then the submodule returns "xyz".

|  |  |  |
| --- | --- | --- |
| **Category** | **Test Data** | **Actual Output** |
| Str1, str2 | paraphrase, paragraph | para |
| Str1, str2 | Beautiful, ugly | “ “ (empty string) |
| Str1, str2 | Phone, phone | phone |
| Str1, str2 | “ “ , “ “ | “ “ (empty string) |
| Str1, str2 | “ hot”, “ “ | “ “ = (empty string) |

1. **Boundary Value Analysis**
2. Apply BVA to the calcGrade submodule from the previous question.

Submodule calcGrade

Imports: mark (integer)

Exports: grade (string)

Calculates a grade, given a mark. For marks less than 50, the grade is "F". For marks from 50 to 59, the grade is "5". For 60 to 69, the grade is "6", and so on up to "10". If mark is invalid, calcGrade will export the empty string "".

Note: invalid is Mark < 0 or Mark > 100

|  |  |  |
| --- | --- | --- |
| Boundaries | Test Data | Expected Result |
| Invalid/”F” | Item = -1  Item = 30 | “ “ (empty)  “F” |
| “F”/”5” | Item = 48  Item = 53 | “F”  “5” |
| “5”/”6” | Item = 56  Item = 65 | “5”  “6” |
| “6”/”7” | Item = 68  Item = 79 | “6”  “7” |
| “7”/”8” | Item = 73  Item = 84 | “7”  “8” |
| “8”/”9” | Item = 82  Item = 94 | “8”  “9” |
| “9”/”10” | Item = 99  Item = 100 | “9”  “10” |
| “10”/Invalid | Item = 100  Item = 103 | “10”  “ “ (empty) |

1. Apply BVA to the following submodule:

Submodule uvRating

Imports: index (real)

Exports: rating (string)

Determines a rating for ultraviolet radiation risk, based on a real-valued UV index. Ratings below zero are invalid, in which case the submodule returns "-". Otherwise, if the index is below 3, the rating is "low", then up to 6 for "moderate", up to 8 for "high", and up to 11 for "very high". Any rating over 11 is "extreme".

Invalid: < 0, return –

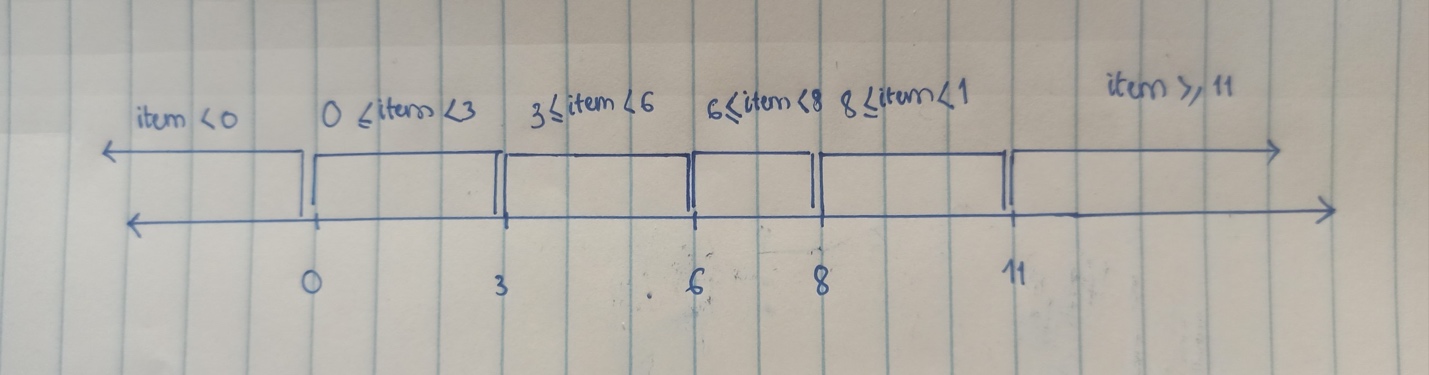
Low: 0 <= i < 3

Moderate: 3 <= i < 6

High: 6 <= i < 8

Very high: 8 <= i < 11

Extreme: i > 11



|  |  |  |
| --- | --- | --- |
| Boundaries | Test Data | Expected Result |
| Invalid/0 | Item = -0.0001  Item = 0 | “-“  “Low” |
| 2.999999/3 | Item = 2.999999  Item = 3 | “Low”  “Moderate” |
| 5.999999/6 | Item = 5.999999  Item = 6 | “Moderate”  “High” |
| 7.999999/8 | Item = 7.999999  Item = 8 | “High”  “Very High” |
| 10.999999/11 | Item = 10.999999  Item = 11 | “Very High”  “Extreme” |