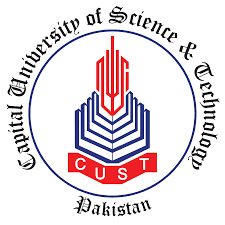
**Group Members**

**BSE181034 Jibran Ahmed**

**BSE181041 Hassan Sheikh**

**ASSIGNMENT 3**

****

**SOFTWARE QUALITY ENGINEERING**

**SUBMITTED TO: Samir Obaid**

Gitlink:

https://github.com/BSE181034/Software-Quality-eng/blob/main/BSE181034.docx

Contents

[Changing in this assignment according to the feedback of assignment 2: 3](#_Toc55254243)

[Function 1 3](#_Toc55254244)

[Function: 3](#_Toc55254245)

[Requirements: 3](#_Toc55254246)

[Causes and effects: 3](#_Toc55254247)

[F1 Graph: 3](#_Toc55254248)

[Function 2 4](#_Toc55254249)

[Function: 4](#_Toc55254250)

[Requirements: 4](#_Toc55254251)

[Causes and effects: 4](#_Toc55254252)

[F1 Graph: 4](#_Toc55254253)

[Function 3 5](#_Toc55254254)

[Function: 5](#_Toc55254255)

[Requirements: 5](#_Toc55254256)

[Causes and effects: 5](#_Toc55254257)

[F1 Graph: 5](#_Toc55254258)

[Justification for choosing EQP: 6](#_Toc55254259)

# Changing in this assignment according to the feedback of assignment 2:

* **String type parameters will not be taken in functions**

# Function 1

## Function:

Void Select\_test\_level (int level , int subject\_no)

## Requirements:

* If level < 0 and subject\_no =1 test will not be open.
* If level = 0 and subject\_no =1 test will not be open.
* If level =1 and subject\_no =1 test will be open.
* If level > 0 and level <11 and subject\_no > 0 and subject\_no < 3 test will be open.

## Causes and effects:

C1 : Level = 1

C2 : Level < 0

C3 : Level > 0 and level < 11

C4 : subject\_no = 1

C5 : subject\_no < 0

C6 : subject\_no > 0 and subject\_no < 3

E1 : Open

E2 : Not opend

## F1 Graph:

**C1**

**C2 E1**

**C3**

**C4**

**C5 E2**

**C6**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 4 |
| Condition/Clause | C1 : Level = 1 | 1 | 0 | 1 | 0 |
| Condition/Clause | C2 : Level < 0 | 0 | 0 | 0 | 1 |
| Condition/Clause | C3 : Level > 0 and level < 11 | 0 | 1 | 0 | 0 |
| Condition/Clause | C4 : subject\_no = 1 | 1 | 0 | 0 | 1 |
| Condition/Clause | C5 : subject\_no < 0 | 0 | 0 | 1 | 0 |
| Condition/Clause | C6 : subject\_no > 0 and subject\_no < 3 | 0 | 1 | 0 | 0 |
| Action/Effort | E1 : Open | + | + | - | - |
| Action/Effort | E2 : Not opend | - | - | + | + |

# Function 2

## Function:

Void Record ( pre\_edu , int age)

## Requirements:

* If Pre\_edu = yes and Age < 3 Not eligible.
* If Pre\_edu = no and Age = 1 Not eligible.
* If Pre\_edu = yes and Age = 3 child is eligible.
* If Pre\_edu = yes and Age => 3 child will be eligible.

## Causes and effects:

C1 : Pre\_edu = Yes

C2 : Pre\_edu = no

C3 : Age < 3

C4 : Age >= 3

C5 : Age = 1

C6 : Age = 3

E1 : Eligible

E2 : Not eligible

## F1 Graph:

**C1**

**C2 E1**

**C3**

**C4**

**C5 E2**

**C6**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 4 |
| Condition/Clause | C1 : Pre\_edu = Yes | 1 | 0 | 1 | 0 |
| Condition/Clause | C2 : Pre\_edu = no | 0 | 0 | 0 | 1 |
| Condition/Clause | C3 : Age < 3 | 0 | 1 | 0 | 0 |
| Condition/Clause | C4 : Age >= 3 | 1 | 0 | 0 | 1 |
| Condition/Clause | C5 : Age = 1 | 0 | 0 | 1 | 0 |
| Condition/Clause | C6 : Age = 3 | 0 | 1 | 0 | 0 |
| Action/Effort | E1 : Eligible | + | + | - | - |
| Action/Effort | E2 : Not eligible | - | - | + | + |

# Function 3

## Function:

Void end\_class( sub, level , final\_score)

## Requirements:

* If sub = 3 and level = 10 and final\_score = 30 child still needs to learn.
* If sub = 2 and level = 9 and final\_score = 30 child still needs to learn
* If sub = 3 and level = 10 and final\_score = 90 child passed the session
* If sub > 2 and level = 10 and final\_score > 89 child passed the session.

## Causes and effects:

C1 : sub = 3

C2 : sub = 2

C3 : level = 9

C4 : level = 10

C5 : Final\_score = 30

C6 : Final\_score = 90

E1 : Pass

E2 : Not Pass

## F1 Graph:

**C1**

**C2 E1**

**C3**

**C4**

**C5**

**C6 E2**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | 1 | 2 | 3 | 4 |
| Condition/Clause | C1 : sub = 3 | 1 | 0 | 1 | 0 |
| Condition/Clause | C2 : sub = 2 | 0 | 0 | 0 | 1 |
| Condition/Clause | C3 : level = 9 | 0 | 1 | 0 | 0 |
| Condition/Clause | C4 : level = 10 | 1 | 0 | 0 | 1 |
| Condition/Clause | C5 : Final\_score = 30 | 0 | 0 | 1 | 0 |
| Condition/Clause | C6 : Final\_score = 90 | 0 | 1 | 0 | 0 |
| Action/Effort | E1 : Pass | + | + | - | - |
| Action/Effort | E2 : Not Pass | - | - | + | + |

# Justification for choosing EQP:

I have used the EQP technique to find out the test cases because in EQP we knows the output by knowing the input and we can test the every possible output of corresponding input as in this case I had applied robust EQP.

Major reason of not choosing BVA is that we do not have any idea that the given input will generate which kind of output.

\_+\_+\_+\_+\_+\_+\_+\_+\_