# Black Box Seminar Problems

1. For each of the following program specifications design test cases using the boundary value approach:

1. If the input value of n is <0, then an appropriate error message must be printed. If 0<=n<20, then the exact value of n! will be printed. If 20<=n<=200, then an approximate value of n! will be printed. Finally if n>200 then the input can be rejected and an error message printed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input Condition** | **Valid Equivalence Class** | **Valid Boundaries** | **Invalid Equivalence Class** | **Invalid Boundaries** |
| **N** |  |  |  |  |
| **value** | **0 - 19** | **0 (1) 19(2)** | **less than 0** | **-1(6)** |
|  | **20- 200** | **20(3) 200(4)** | **greater than 200** | **201(7)** |
|  | **integer(5)** |  | **not an integer(8)** |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Output Condition** | **Valid Equivalence Class** | **Valid Boundaries** | **Invalid Equivalence Class** | **Invalid Boundaries** |
| **error messages** |  |  |  |  |
| **value** | **-ve message(9)** |  |  |  |
|  | **too big message(10)** |  |  |  |

Valid Test Cases Boundaries Covered

0 1,5

19 2,5

20 3,5

200 4,5

Invalid Test Cases

-1 6

201 7

2.0 8

2. The input to the program consists of a student’s name which may be up to 20 letters in length, a set of up to three marks each an integer in the range 1 to 100. There will be at least 1 mark for a student. The output from the program will be a list of the students and their average mark.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input Condition** | **Valid Equivalence Class** | **Valid Boundaries** | **Invalid Equivalence Class** | **Invalid Boundaries** |
| **Name** |  |  |  |  |
| **size** | **1-20** | **1(1) 20(2)** | **<1, >20** | **0(12) 21(13)** |
| **content** | **letters** | **‘A’(3) ‘Z’(4) ‘a’(5) ‘z’(6)** | **non letters(14)** |  |
| **Marks** |  |  |  |  |
| **value** | **1-100** | **1(7) 100(8)** | **<1, >100** | **0(15) 101(16)** |
| **type** | **integer(9)** |  | **not integer(17)** |  |
| **number of** | **1-3** | **1(10) 3(11)** | **<1, >3** | **0(18) 4(19)** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Output Condition** | **Valid Equivalence Class** | **Valid Boundaries** | **Invalid Equivalence Class** | **Invalid Boundaries** |
| **List** |  |  |  |  |
| **size** | **1-Max** | **1(20) Max(21)** | **<1, >Max** | **0(24) Max+1(25)** |
| **Ave. Mark** |  |  |  |  |
| **value** | **1-100** | **1(22) 100(23)** | **<1, >100** | **0(26) 101(27)** |

Valid Test Cases Boundaries Covered

A 1 (1),(3),(7),(9),(10)

Zazefghijklmnopqrstu 100 45 23 (2),(4),(5),(6),(8),(9),(11)

A 1 1 1 (20),(22) // 1 student with avg of 1

max number of entries of form

a 100 100 100 (21),(23) //max students with avg of 100

Invalid Test Cases

1 (12)

abcdefghijklmnopqrstu 100 45 23 (13)

fred5 37 45 (14)

fred 0 (15)

fred 101 (16)

fred \* (17)

fred (18)

fred 1 2 3 4 (19)

no data - just a return or empty file (24)

max number+1 of entries of form (25)

a 100 100 100

a 0 0 0 (26)

a 101 101 101 (27)

3. Input to a tax calculation program consists of an identification number, ( an upper case letter followed by 1 to 5 digits), a value for marital status, (i.e. single, married, divorced, widowed, separated), and a gross salary. The tax charged is based on three bands:- 1-5000, 5001-20000, 20000+. The tax free allowance is 5000. The tax rate calculated also depends on the marital status. (For this example ignore output classes)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input Conditions** | **Valid Equivalence Classes** | **Valid Boundaries** | **Invalid Equivalence Classes** | **Invalid Boundaries** |
| **ID** |  |  |  |  |
| **size** | **2-6** | **2(1) 6(2)** | **<2, >6** | **1(18) 7(19)** |
| **1st character** | **letter** | **‘A’(3) ‘Z’(4)** | **not a letter(20)** |  |
| **remaining characters** | **digits** | **‘0’(5) ‘9’(6)** | **not digits(21)** |  |
| **Status** |  |  |  |  |
| **value** | **single(7)** |  | **other(22)** |  |
|  | **married(8)** |  |  |  |
|  | **divorced(9)** |  |  |  |
|  | **widowed(10)** |  |  |  |
|  | **separated(11)** |  |  |  |
| **Salary** |  |  |  |  |
| **value (need to know tax allowance, say 5000)** | **1-5000** | **1(12) 5000(13)** | **<1** | **0(23)** |
|  | **5001 - 20000** | **5001(14)**  **20000(15)** |  |  |
|  | **>20000** | **20001(16)** |  |  |
| **type** | **integer(17)** |  | **not integer(24)** |  |

In order to derive the test cases for this problem we need to know a little more about the operation of the program. Are the various categories charged at the same tax rate in each band or not., is the tax allowance a constant for each class? For ease we will assume a constant tax allowance but assume that each category is charged at a different rate.

Valid Test Cases Equivalence Classes Covered

A0 single 1 (1),(3),(5),(7),(12),(17)

Z99999 single 5000 (2),(4),(6),(7),(13) ,(17)

A9 single 5001 (1),(3),(6),(7),(14) ,(17)

A00000 single 20000 (2),(3),(5),(7),(15) ,(17)

A00000 single 20001 (2),(3),(5),(7),(16) ,(17)

A0 married 1 (1),(3),(5),(8),(12) ,(17)

A99999 married 5000 (2),(3),(6),(8), (13) ,(17)

A9 married 5001 (1),(3),(6),(8),(14) ,(17)

A00000 married 20000 (2),(3),(5),(8),(15) ,(17)

A00000 married 20001 (2),(3),(5),(8),(16) ,(17)

A0 divorced 1 (1),(3),(5),(9),(12) ,(17)

A99999 divorced 5000 (2),(4),(6),(9),(13) ,(17)

A9 divorced 5001 (1),(4),(6),(9),(14) ,(17)

A00000 divorced 20000 (2),(3),(5),(9),(15) ,(17)

A00000 divorced 20001 (2),(3),(5),(9),(16) ,(17)

A0 widowed 1 (1),(3),(5),(10),(12) ,(17)

A99999 widowed 5000 (2),(4),(6),(10),(13) ,(17)

A9 widowed 5001 (1),(3),(6),(10),(14) ,(17)

A00000 widowed 20000 (2),(3),(5),(10),(15) ,(17)

A00000 widowed 20001 (2),(3),(5),(10),(16) ,(17)

A0 separated 1 (1),(3),(5),(11),(12) ,(17)

A99999 separated 5000 (2),(4),(6),(11),(13) ,(17)

A9 separated 5001 (1),(3),(6),(11),(14) ,(17)

A00000 separated 20000 (2),(3),(5),(11),(15) ,(17)

A00000 separated 20001 (2),(3),(5),(11),(16) ,(17)

Invalid Test Cases

a single 5000 (18)

c1234567 single 5000 (19)

\* single 10000 (20)

D$ single 30000 (21)

A12345 worried 1234 (22)

B123 single 0 (23)

C123 single @ (24)

4. f the tariff code is a B then fuel costs are charged at 15p/unit for the first 200 units and 5p/unit for the rest. When the tariff code is H the charges are 10p/unit for the first 100 units and 8p/unit for the rest. The program input is a code followed by a number of units. The program output will be the code, the number of units, the cost for the fuel. If an invalid code is entered or if the number of units is greater than 9999 an appropriate error message will be output.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input Condition** | **Valid Equivalene Class** | **Valid Boundaries** | **Invalid Equivalene Class** | **Invalid Boundaries** |
| **Code** |  |  |  |  |
| **value** | **B,H** | **B(1)  H(2)** | **other(12)** |  |
| **Units** |  |  |  |  |
| **value** | **0-200** | **0(3)  200(4)** | **<0** | **-1(13)** |
|  | **201-9999** | **201 (5)**  **9999(6)** |  | **10000(14)** |
|  | **0-100** | **0(7)  100(8)** |  |  |
|  | **101-9999** | **101(9)**  **9999(10)** |  |  |
| **type** | **integer(11)** |  | **not integer(15)** |  |

We need the 4 subranges for Units because the specification indicates that the program will behave differently for the ranges depending on the value of Code.

Valid Test Cases Equivalence Classes Covered

B 0 (1),(3),(11)

B 200 (1),(4),(11)

B 201 (1),(5),(11)

B 9999 (1),(6),(11)

H 0 (2),(7),(11)

H 100 (2),(8),(11)

H 101 (2),(9),(11)

H 9999 (2),(10),(11)

Invalid Test Cases

Z 56 (12)

B -1 (13)

H 10000 (14)

H B (15)

The output analysis would involve us looking at the actual output both valid and error messages. For the valid output we would need some indication of the field sizes for the cost and using this information we would generate cases involving a minimum number of characters, maximum number of characters and maximum plus one. Of the 2 error messages mentioned onewe have already coveredwhile the other suggests we might need a test with the number of units set to 10000. This last case again is picked up in the input analysis.

5. A user can ‘dial’ a bank using his or her personal computer, provide a six-digit password, and follow with a series of keyword commands that can trigger various banking functions. The data is accepted in the following form:

area code - blank or three-digit number; the second digit must be a 0 or 1, may not be the value 606

prefix - three digit number, not beginning with 0 or 1

suffix - four digit number

password - six character alphanumeric value

command - ‘check’, ‘deposit’, ‘pay’

(Ignore output with this problem)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input Condition** | **Valid Equivalence Classes** | **Valid Boundaries** | **Invalid Equivalence Classes** | **Invalid Boundaries** |
| **Area Code** |  |  |  |  |
| **exists** | **blank(1)** |  |  |  |
|  | **present(2)** |  |  |  |
| **size** | **3 chars** | **3(3)** | **<3 , >3** | **2(26) 4(27)** |
| **content** | **digits** | **‘0’(4) ‘9’(5)** | **other(28)** |  |
| **2nd digit** | **{0,1}** | **‘0’(6) ‘1’(7)** | **‘2’(29) ‘9’(30)** |  |
| **Prefix** |  |  |  |  |
| **size** | **3 chars** | **3(8)** | **<3 , >3** | **2(31) 4(32)** |
| **1st character** | **digits -{0,1}** | **‘2’(9) ‘9’(10)** | **‘0’(33), ‘1’(34)** |  |
|  |  |  | **other(35)** |  |
| **rest** | **digits** | **‘0’(11) ‘9’(12)** | **other(36)** |  |
| **Suffix** |  |  |  |  |
| **size** | **4 chars** | **4(13)** | **<4 , >4** | **3(37) 5(38)** |
|  | **digits** | **‘0’(14) ‘9’(15)** | **other(39)** |  |
| **Password** |  |  |  |  |
| **size** | **6 chars** | **6(16)** | **<6 , >6** | **5(40) 7(41)** |
| **content** | **digits** | **‘0’(17) ‘9’(18)**  **‘a’(19) ‘z’(20)**  **‘A’(21) ‘Z’(22)** | **other(42)** |  |
| **Command** |  |  |  |  |
| **value** | **check(23)** |  |  |  |
|  | **deposit(24)** |  |  |  |
|  | **pay(25)** |  | **other(43)** |  |

Valid Test Cases Equivalence Classes Covered

290 0967 090909 check (1),(6),(7),(9),(10),(11),(12),

(13),(14),(15),(16),(17)

902 909 9090 909090 deposit (2), (3),(4) (5), (6),(8),(9),(10),

(11),(12),(13) (14),(15),(16),(18)

902 909 9090 909090 pay (2), (3),(4) (5), (6),(8),(9),(10),

(11),(12),(13) (14),(15),(16),(18)

Invalid Test Cases

90 909 9090 909090 deposit (20)

9023 909 9090 909090 deposit (21)

abc 909 9090 909090 deposit (22)

902 90 9090 909090 deposit (23)

902 9090 9090 909090 deposit (24)

902 009 9090 909090 deposit (25)

902 109 9090 909090 deposit (26)

902 a09 9090 909090 deposit (27)

902 9+9 9090 909090 deposit (28)

902 909 909 909090 deposit (29)

902 909 90900 909090 deposit (30)

902 90 9 %090 909090 deposit (31)

902 90 9 9090 90909 deposit (32)

902 90 9 9090 9090909 deposit (33)

902 90 9 9090 $09090 deposit (34)

902 90 9 9090 909090 other (35)