

Ingeniería del Software 2 – Segundo Ejercicio Teoría

Tercer Problema

Autores:

- Aitor Jorge Jiménez.
- Óscar Veloso Delgado.



Índice

1. Código.
2. Variables.
3. Casos de prueba máximos.
4. Casos de prueba para cumplimentar each use.
5. Casos de prueba para alcanzar cobertura pairwise.
6. Comentar resultados apartados 4 y 5.

1. Código.

Main.java

```
Main.java ×
1 package C03_Testing_P3;
2
3 public class Main{
4
5     public static void main(String[] args) {
6
7         Adecuacion_Funcional adecuacionFun = new Adecuacion_Funcional(0,0,0,80);
8         mantenibilidad mantenibilidad = new mantenibilidad(0,0,0,0,0,90);
9         Certificado certificadofin = new Certificado(adecuacionFun, mantenibilidad);
10        System.out.println(certificadofin.certificadofin());
11    }
12 }
```

Adeucacion_Funcional.java

```
Adeucacion_Funcional.java ×
1 package C03_Testing_P3;
2
3 public class Adeucacion_Funcional {
4
5     int adecuacionFun;
6
7     int completitudFun;
8     int correccionFun;
9     int pertinenciaFun;
10
11     int rango;
12
13     public Adeucacion_Funcional(int completitudFun, int correccionFun, int pertinenciaFun, int rango) {
14
15         this.completitudFun = completitudFun;
16         this.correccionFun = correccionFun;
17         this.pertinenciaFun = pertinenciaFun;
18         this.rango = rango;
19     }
20
21     // Metodos necesarios para adecuacion
22
23     public int medicionCompletitudFun(int rango) {
24
25         if(rango>=0 && rango<10) {
26
27             completitudFun= 0;
28         }else if(rango>=10 && rango<35) {
29
30             completitudFun= 1;
31         }else if(rango>=35 && rango<50) {
32
33             completitudFun= 2;
34
35         }else if(rango>=50 && rango<70) {
36
37             completitudFun= 2;
38         }else if(rango>=70 && rango<90) {
39
40             completitudFun= 3;
41         }else if(rango>=90 && rango<100) {
42             completitudFun= 4;
43         }
44
45         return completitudFun;
46     }
47
48     public int medicionCorreccionFun(int rango) {
49
50         if(rango>=0 && rango<10) {
51             correccionFun= 0;
```

```

52     }
53     else if(rango>=10 && rango<35) {
54         correccionFun= 1;
55     }
56     else if(rango>=35 && rango<50) {
57         correccionFun= 2;
58     }
59     else if(rango>=50 && rango<70) {
60         correccionFun= 2;
61     }
62     else if(rango>=70 && rango<90) {
63         correccionFun= 3;
64     }
65     else if(rango>=90 && rango<100) {
66         correccionFun= 4;
67     }
68
69     return correccionFun;
70 }
71
72 public int medicionPertinencia(int rango) {
73
74     if(rango>=0 && rango<10) {
75
76         pertinenciaFun= 0;
77     }
78     else if(rango>=10 && rango<35) {
79
80         pertinenciaFun= 1;
81     }
82     else if(rango>=35 && rango<50) {
83
84         pertinenciaFun= 2;
85     }
86     else if(rango>=50 && rango<70) {
87         pertinenciaFun= 2;
88     }
89     else if(rango>=70 && rango<90) {
90         pertinenciaFun= 3;
91     }
92     else if(rango>=90 && rango<100) {
93         pertinenciaFun= 4;
94     }
95
96     return pertinenciaFun;
97 }
98
99
100 public int AdecuacionFuncional(int a, int b,int c) {
101
102     int min= 10;
103     int num= 0;
104     int[] mediciones = (new int[] {a,b,c});
105
106     for(int i =0; i<mediciones.length; i++) {
107
108         num= mediciones[i];
109
110         if (num<min) {
111             min = num;
112         }
113     }
114
115     adecuacionFun = min;
116     return adecuacionFun;
117 }
118 }
119

```

Mantenibilidad.java

```

1 package C03_Testing_P3;
2
3 public class mantenibilidad {
4
5     int mantenibilidad;
6
7     int modularidad;
8     int reusabilidad;
9     int analizabilidad;
10    int capacidadMod;
11    int capacidadPro;
12
13    int rango;
14
15    public mantenibilidad(int modularidad, int reusabilidad,int analizabilidad, int capacidadMod,
16        int capacidadPro, int calidadFinal, int rango) {
17
18        this.modularidad = modularidad;
19        this.reusabilidad = reusabilidad;
20        this.analizabilidad = analizabilidad;
21        this.capacidadMod = capacidadMod;
22        this.capacidadPro = capacidadPro;
23        this.rango = rango;
24    }
25

```

```

26 public int medicionModularidad(int rango) {
27
28     if(rango>=0 && rango<10) {
29         modularidad= 0;
30     }
31     if(rango>=10 && rango<35) {
32         modularidad= 1;
33     }
34     if(rango>=35 && rango<50) {
35         modularidad= 2;
36     }
37     if(rango>=50 && rango<70) {
38         modularidad= 2;
39     }
40     if(rango>=70 && rango<90) {
41         modularidad= 3;
42     }
43     if(rango>=90 && rango<100) {
44         modularidad= 4;
45     }
46
47     return modularidad;
48 }
49
50 public int medicionReusabilidad(int rango) {
51
52     if(rango>=0 && rango<10) {
53         reusabilidad= 0;
54     }
55     if(rango>=10 && rango<35) {
56         reusabilidad= 1;
57     }
58     if(rango>=35 && rango<50) {
59         reusabilidad= 2;
60     }
61     if(rango>=50 && rango<70) {
62         reusabilidad= 2;
63     }
64     if(rango>=70 && rango<90) {
65         reusabilidad= 3;
66     }
67     if(rango>=90 && rango<100) {
68         reusabilidad= 5;
69     }
70
71     return reusabilidad;
72 }
73 public int medicionAnalizabilidad(int rango) {
74
75     if(rango>=0 && rango<10) {
76         analizabilidad= 0;
77     }
78     if(rango>=10 && rango<35) {
79         analizabilidad= 0;
80     }
81     if(rango>=35 && rango<50) {
82         analizabilidad= 1;
83     }
84     if(rango>=50 && rango<70) {
85         analizabilidad= 2;
86     }
87     if(rango>=70 && rango<90) {
88         analizabilidad= 3;
89     }
90     if(rango>=90 && rango<100) {
91         analizabilidad= 5;
92     }
93
94     return analizabilidad;
95 }
96 public int medicionCapacidadMod(int rango) {
97
98     if(rango>=0 && rango<10) {
99         capacidadMod= 0;
100     }
101     if(rango>=10 && rango<35) {
102         capacidadMod= 1;
103     }
104     if(rango>=35 && rango<50) {
105         capacidadMod= 2;
106     }
107     if(rango>=50 && rango<70) {
108         capacidadMod= 3;
109     }
110     if(rango>=70 && rango<90) {
111         capacidadMod= 4;
112     }
113     if(rango>=90 && rango<100) {
114         capacidadMod= 5;
115     }
116
117     return capacidadMod;
118 }
119 public int medicionCapacidadPro(int rango) {
120
121     if(rango>=0 && rango<10) {
122         capacidadPro= 0;
123     }

```

```

124     if(rango>=10 && rango<35) {
125         capacidadPro= 1;
126     }
127     if(rango>=35 && rango<50) {
128         capacidadPro= 1;
129     }
130     if(rango>=50 && rango<70) {
131         capacidadPro= 2;
132     }
133     if(rango>=70 && rango<90) {
134         capacidadPro= 4;
135     }
136     if(rango>=90 && rango<100) {
137         capacidadPro= 4;
138     }
139
140     return capacidadMod;
141 }
142
143
144 public int Mantenibilidad(int modularidad, int reusabilidad, int analizabilidad, int capacidadMod, int capacidadPro) {
145     int min= 10;
146     int num= 0;
147     int[] mediciones = (new int[]) {modularidad, reusabilidad, analizabilidad, capacidadMod, capacidadPro};
148
149     for(int i =0; i<mediciones.length; i++) {
150         num= mediciones[i];
151         if (num<min) {
152             min = num;
153         }
154     }
155     mantenibilidad = min;
156     return mantenibilidad;
157 }
158 }
159

```

Certificado.Java

```

Certificado.java ×
1 package C03_Testing_P3;
2
3 public class Certificado{
4
5
6     mantenibilidad mantenibilidad;
7     Adecuacion_Funcional adecuacionFun;
8
9
10    public Certificado(Adecuacion_Funcional adecuacionFun, mantenibilidad mantenibilidad) {
11        this.adecuacionFun = adecuacionFun;
12        this.mantenibilidad = mantenibilidad;
13    }
14
15    public String certificadoFin() {
16        int certificadoFinal=0;
17        String msgCertificado= "";
18
19        int adecuacion = adecuacionFun.AdecuacionFuncional(adecuacionFun.medicionCompleitudFun(adecuacionFun.rango)
20            ,adecuacionFun.medicionCorreccionFun(adecuacionFun.rango), adecuacionFun.medicionPertinencia(adecuacionFun.rango));
21
22        int mantenibilidadFinal = mantenibilidad.Mantenibilidad(mantenibilidad.medicionModularidad(mantenibilidad.rango), mantenibilidad.medicionReusabilidad(mantenibilidad.rango),
23            mantenibilidad.medicionAnalizabilidad(mantenibilidad.rango), mantenibilidad.medicionCapacidadMod(mantenibilidad.rango), mantenibilidad.medicionCapacidadPro(mantenibilidad.rango));
24
25        if (adecuacion == 1) {
26
27            if (mantenibilidadFinal ==1) {
28                certificadoFinal = 1;
29            }else if (mantenibilidadFinal ==2) {
30                certificadoFinal = 1;
31            }else if (mantenibilidadFinal ==3) {
32                certificadoFinal = 1;
33            }else if (mantenibilidadFinal ==4) {
34                certificadoFinal = 1;
35            }else if (mantenibilidadFinal ==5) {
36                certificadoFinal = 1;
37            }
38        }else if (adecuacion == 2) {
39
40            if(mantenibilidadFinal==1) {
41                certificadoFinal = 1;
42            }else if (mantenibilidadFinal ==2) {
43                certificadoFinal = 2;
44            }else if (mantenibilidadFinal ==3) {
45                certificadoFinal = 2;
46            }else if (mantenibilidadFinal ==4) {
47                certificadoFinal = 2;
48            }else if (mantenibilidadFinal ==5) {
49                certificadoFinal = 2;
50            }
51        }else if (adecuacion==3) {

```

```

52         if(mantenibilidadFinal==1) {
53             certificadoFinal =2;
54         }else if (mantenibilidadFinal ==2) {
55             certificadoFinal = 2;
56         }else if (mantenibilidadFinal ==3) {
57             certificadoFinal = 3;
58         }else if (mantenibilidadFinal ==4) {
59             certificadoFinal = 3;
60         }else if (mantenibilidadFinal ==5) {
61             certificadoFinal = 3;
62         }
63     }else if (adecuacion == 4) {
64
65         if (mantenibilidadFinal ==1) {
66             certificadoFinal =3;
67         }else if (mantenibilidadFinal==2) {
68             certificadoFinal = 3;
69         }else if (mantenibilidadFinal==3) {
70             certificadoFinal = 3;
71         }else if (mantenibilidadFinal==4) {
72             certificadoFinal = 3;
73         }else if (mantenibilidadFinal ==5) {
74             certificadoFinal = 4;
75         }
76     }else if(adecuacion==5) {
77
78         if (mantenibilidadFinal==1) {
79             certificadoFinal = 3;
80         }else if (mantenibilidadFinal==2) {
81             certificadoFinal = 3;
82         }else if (mantenibilidadFinal ==3) {
83             certificadoFinal = 4;
84         }else if (mantenibilidadFinal ==4) {
85             certificadoFinal = 4;
86         }else if (mantenibilidadFinal==5) {
87             certificadoFinal = 5;
88         }
89     }
90
91     if (certificadoFinal >= 3) {
92
93         msgCertificado= "El software se ha certificado con una calidad de "+certificadoFinal;
94     }else {
95         msgCertificado = "No se ha podido certificar el software" +certificadoFinal;
96     }
97
98     return msgCertificado;
99 }
100 }
101

```

2. Variables

Variable	Tipo	Rango
adecuacionFun	int	(-32768,32767)
completitudFun	int	(-32768,32767)
correccionFun	int	(-32768,32767)
pertinenciaFun	int	(-32768,32767)
rango	int	(-32768,32767)
mantenibilidad	int	(-32768,32767)
modularidad	int	(-32768,32767)
reusabilidad	int	(-32768,32767)
analizabilidad	int	(-32768,32767)
capacidadMod	int	(-32768,32767)
capacidadPro	int	(-32768,32767)

3. Casos de prueba máximos.

- Rango Mediciones Adecuación = 5
 - Variantes completitud Funcional = 5
 - Variantes corrección funcional = 5
 - Variantes pertinencia Funcional= 5
-
- Rango mediciones mantenibilidad =5
 - Variantes modularidad = 5
 - Variantes reusabilidad = 5
 - Variante analizabilidad = 5
 - Variante Capacidad de ser modificado = 5
 - Variante Capacidad de ser probado = 5
-
- Variantes finales de A.Funcional = 5
 - Variantes finales de Mantenibilidad = 5
- Total = $(5 \times 5 \times 5 \times 5) \times (5 \times 5 \times 5 \times 5 \times 5 \times 5) \times (5 \times 5) = 244.140.625$ casos de prueba diferentes.

4. Casos de prueba para cumplimentar each use.

- CertificadoFin()

Adecuación	Mantenibilidad	Certificado
1	0	Null
2	2	False
4	4	True

5. Casos de prueba para alcanzar cobertura pairwise.

Se ha cubierto el 88,6%, ya que con el valor de la tabla de adecuación no se podrá acceder al valor 5 de esta debido a que su valor mínimo siempre va a ser 4.

RANGO Adecuación	RANGO Mantenibilidad	ADECUACION	MANTENIBILIDAD	CERTIFICADO
[0,10) - 0	[0-10) - 0	0	0	False
[10-35) - 22	[10-35) - 15	1	0	False
[35-50) - 40	[35-50) - 40	1	1	False
[50-70) - 55	[50-70) - 60	2	2	False
[70-90) - 80	[70-90) - 85	3	3	True
[90-100) - 91	[90-100) - 95	4	4	True
[35-50) - 40	[70-90) - 70	1	3	False

6. Comentar resultados apartados 4 y 5

Una vez realizado el Testing en Eclipse con Junit, hemos obtenido una cobertura del 88,6%, es decir, hemos cubierto 952 instrucciones de las 1074 que contiene. Se ha generado el siguiente informe con surefire:

Project Documentation

- Project Reports
 - Surefire Report**



Surefire Report

Summary

[Summary] [Package List] [Test Cases]

Tests	Errors	Failures	Skipped	Success Rate	Time
1	0	0	0	100%	0.04

Note: failures are anticipated and checked for with assertions while errors are unanticipated.

Package List

[Summary] [Package List] [Test Cases]

Package	Tests	Errors	Failures	Skipped	Success Rate	Time
C03_Testing_P3	1	0	0	0	100%	0.04

Note: package statistics are not computed recursively, they only sum up all of its test suites numbers.

C03_Testing_P3

Class	Tests	Errors	Failures	Skipped	Success Rate	Time
 CertificadoTest	1	0	0	0	100%	0.04

Test Cases

[Summary] [Package List] [Test Cases]

CertificadoTest

	testCertificadoIn1	0.003
---	--------------------	-------

Copyright © 2014 All rights reserved.