PSP1.0 Ejercicio 4A

1. Resumen del plan del proyecto nivel PSP1.0

PSP1 Project Plan Summary - Program 4A

Student	1-Rubén Ignacio Couoh Ku		Date	02/02/2017
Program	Linear Regression		Program#	4A
Instructor	Carlos Mojica		Language	NodeJS v6.9.4

	Plan	Actual	To Date	To Date%
Summary				
LOC/Hour	50	57	62	
Program Size (LOC)				
Base(B)	**128**	**146**		
Deleted(D)	**0**	**0**		
Modified(M)	**8**	**16**		
Added(A)	49	43		
Reused(R)	**0**	**0**	0	
Total N&C (N)	**57**	59	387	
Total LOC(T)	177	189	583	
Total New Reused	**0**	**0**	26	
Total Object LOC(E)	**58**	**65**	65	
Time in Phase (min.)				
Planning	6	14	42	11.0
Design	2	10	20	5.3
Code	44	28	225	59.1
Compile	4	2	20	5.3
Test	6	7	35	9.1
Postmortem	6	11	39	10.2
Total	**70**	71	381	100.0
Defects Injected				
Planning		0	0	0.0
Design		0	0	0.0
Code		1	13	100.0
Compile		0	0	0.0
Test		0	0	0.0
Total Development		1	13	100.0
Defects Removed				
Planning		0	0	0.0
Design		0	0	0.0
Code		0	0	0.0
Compile		1	11	84.6
Test		0	2	15.4
Total Development		1	13	100.0
After Development		0	0	

2. Resumen del plan del proyecto nivel PSP0.1 3A

PSP0.1 Project Plan Summary - Program 3A

Student	1-Rubén Ignacio Couoh Ku		Date	26/01/2017
Program	Object Counter		Program#	3A
Instructor	Carlos Mojica		Language	NodeJS v6.9.4

	Plan	Actual	To Date	To Date%
Program Size (LOC)				
Base(B)		73		
Deleted(D)		2		
Modified(M)		5		
Added(A)		55		
Reused(R)		0	0	
Total N&C (N)	65	60	328	
Total LOC(T)		126	394	
Total New Reused		26	26	
Time in Phase (min.)				
Planning	8	9	28	9.0
Design	2	5	10	3.2
Code	69	43	197	63.8
Compile	8	1	19	6.0
Test	11	4	28	8.9
Postmortem	7	14	28	9.1
Total	105	76	309	100.0
Defects Injected				
Planning		0	0	0.0
Design		0	0	0.0
Code		1	12	100.0
Compile		0	0	0.0
Test		0	0	0.0
Total Development		1	12	100.0
Defects Removed				
Planning		0	0	0.0
Design		0	0	0.0
Code		0	0	0.0
Compile		1	10	83.3
Test		0	2	16.7
Total Development		1	12	100.0
After Development		0	0	

3. Reporte de pruebas.

Test Name/Number	1
Test Objective	Calcular los parámetros de regresión B0 y B1 de una colección de pares ordenados.
Test Description	Calcular los parámetros de regresión B0 y B1 de una colección de pares ordenados que se encuentran guardados en un archivo de texto de acuerdo con las siguientes condiciones:
	1 Cada línea del archivo representa un par ordenado.
	2 Los valores de cada línea del archivo se encuentran separados por un espacio.
	3 En una línea solamente se puede guardar 2 valores (valorX ValorY)
	Para esta prueba se utilizara una colección de 10 elementos donde los valores de las Xs serán las LOC estimadas de los objetos y para los valores de las Ys se utilizaran las LOC nuevas y cambiadas actuales. Entrada: 130 186 650 699 99 132 150 272 128 291 302 331 95 199 945 1890 368 788 961 1601
Test Conditions	NodeJS v6.4.9
Expected Results	B0: -22.55 B1: 1.7279
Actual Results	B0: -22.55 B1: 1.7279

Test Name/Number	2
Test Objective	Calcular los parámetros de regresión B0 y B1 de una colección de pares ordenados.
Test Description	Calcular los parámetros de regresión B0 y B1 de una colección de pares ordenados que se encuentran guardados en un archivo de texto de acuerdo con las siguientes condiciones: 1 Cada línea del archivo representa un par ordenado. 2 Los valores de cada línea del archivo se encuentran separados por un espacio. 3 En una línea solamente se puede guardar 2 valores (valorX ValorY) Para esta prueba se utilizara una colección de 10 elementos donde los valores de las Xs serán las LOC estimadas de los objetos y para los valores de las Ys se utilizaran las LOC nuevas y cambiadas actuales. Entrada: 163 186 765 699 141 132 166 272 137 291 355 331 136 199 1206 1890 433 788 1130 1601
Test Conditions	NodeJS v6.4.9
Expected Results	B0: -23.92 B1: 1.4310
Actual Results	B0: -23.92 B1: 1.4310

Test Name/Number	3
Test Objective	Calcular los parámetros de regresión B0 y B1 de una colección de pares ordenados.
Test Description	Calcular los parámetros de regresión B0 y B1 de una colección de pares ordenados que se encuentran guardados en un archivo de texto de acuerdo con las siguientes condiciones: 1 Cada línea del archivo representa un par ordenado. 2 Los valores de cada línea del archivo se encuentran separados por un espacio. 3 En una línea solamente se puede guardar 2 valores (valorX ValorY) Para esta prueba se utilizara una colección de 3 elementos donde los valores de las Xs serán las LOC nuevas y cambiadas estimadas y para los valores de las Ys se utilizaran las LOC nuevas y cambiadas actuales de los programas 2A, 3A y 4A. Entrada: 80 83 65 60 57 59
Test Conditions	NodeJS v6.4.9
Expected Results	B0: -7.16 B1: 1.1064
Actual Results	B0: -7.16 B1: 1.1064

4. Forma de registros PIPs.

Propuesta de Mejora de Proceso (PIP)

Nombre: Rubén Ignacio Couoh Ku Fecha: 02/02/2017
Proyecto: 4A
Proceso: PSP1.0 Plantilla de estimación de tamaño

Número de PIP

	Descripción del Problema:
1	La plantilla de estimación de tamaño en la sección de adicionados a la base debería te tener alguna opción para poder indicar que se agregó a la base pero con el objetivo de ser reutilizado. Por ejemplo: si solamente se desea agregar una función matemática a una clase X que se toma como base, no existe la opción para indicar que se desea agregar esa función como reutilizable, sin embargo, se codifico la función con el objetivo de ser reutilizada.

PROPUESTA

# PIP	**Descripción de la Propuesta**
1	Permitir marcar lo que se adiciona la base con un * indicando que se agregó para ser reutilizado de forma similar a como se indica en la sección de nuevos objetos.

5. Plantilla de estimación de tamaño.

Size Estimating Template

Student	Rubén Ignacio Couoh Ku	Date	02/02/2017
Instructor	Carlos Mojica	Program#	4A

BASE PROGRAM LOC				ESTIMATE	ACTUAL
BASE SIZE (B) =>				128	146
LOC DELETED (D) =>				0	0
LOC MODIFIED (M) =>				8	16
OBJECT LOC					
BASE ADDITIONS:	TYPE	METHODS	REL. SIZE	LOC	LOC
Calculo de B0	C	1	S	5.0	8
Calculo de B1	C	1	L	24.0	15
N dados en la Lista	D	1	S	5.0	9
Leer archivo de texto en 2 columna	IO	1	M	16.0	17
(BA) subtotal from page 2				0.0	0
TOTAL BASE ADDITIONS (BA)				50.0	49
NEW OBJECTS:	TYPE	METHODS	REL. SIZE	LOC *	LOC *
(NO) subtotal from page 2				0.0	0
TOTAL NEW OBJECTS (NO)				0.0	0
REUSED OBJECTS				LOC	LOC
(R) subtotal from page 2				0.0	0
REUSED TOTAL (R)				0.0	0

		Size	Time
Estimated Object LOC:	E = BA+NO+M	58	
Regression Parameter:	B_0	0.00	0.00
Regression Parameter:	B_1	0.99	1.20
Estimated New and Changed LOC:	N = B_0 + B_1` * E	57.2	
Estimated Total LOC:	T = N + B - D - M + R	177.2	
Estimated Total New Reuse (sum of * LOC):		0	
Estimated Total Developent Time:	Time = B_0 + B_1` * E		69.7
Prediction Range:	Range	0.0	0.0
Upper Prediction Interval:	UPI = N + Range		
Lower Prediction Interval:	LPI = N - Range		
Prediction Interval Percent		70%	70%
Method Selected		C	C
R^2		0.00	0.00

6. Forma de registro de tiempos.

Project	Phase	Date	Start	Int.	Stop	Delta	Comments
4	PLAN	02/02/17	12:07:31		12:08:39	1.1	Revisión de las formulas para calcular los parámetos de regresión.
4	PLAN	02/02/17	12:11:45		12:15:15	3.5	Entrada de datos por archivos, almacenar los datos en una lista ligada, objeto matemático para calcular los parámetros de la regresión, imprimir los resultados en pantalla, agregar la logica en el programa principal.
4	PLAN	02/02/17	12:54:52		13:04:19	9.4	Estimación de tamaños.
4	DLD	02/02/17	13:46:12		13:56:30	10.3	Diagrama de secuencia B0 y B1
4	CODE	02/02/17	13:55:58		14:04:14	8.3	Codificación de la entrada de datos por archivos.
4	CODE	02/02/17	14:17:34		14:19:56	2.4	Se agregó a la lista ligada la opción para guardar N registros.
4	CODE	02/02/17	14:31:22		14:43:49	12.5	Codificación del método para calcular B1.
4	CODE	02/02/17	14:45:49		14:47:56	2.1	Codificación del método para calcular B0.
4	CODE	02/02/17	14:49:00		14:51:22	2.4	Codificación del programa principal.
4	COMPILE	02/02/17	15:25:11		15:26:58	1.8	Error al inicializar las variables en el cilco for (let Xi = listX.begin(), let Yi = listY.begin(); ...
4	TEST	02/02/17	16:54:29		16:56:56	2.4	Prueba 1 LOC estimadas de los objetos y las LOC nuevas y cambiadas actuales
4	TEST	02/02/17	16:57:07		16:58:37	1.5	Prueba 2 LOC nuevas y cambiadas estimadas con las LOC nuevas y cambiadas actuales
4	TEST	02/02/17	16:58:50		17:01:49	3.0	Prueba 3, 2A, 3A, 4A LOC nuevas y cambiadas estimadas vs. LOC nuevas y cambiadas actuales
4	PM	02/02/17	17:08:41		17:11:07	2.4	Registro de defectos inyectados
4	PM	02/02/17	17:11:53		17:13:02	1.2	Registro de defectos removidos
4	PM	02/02/17	17:14:02		17:17:10	3.1	Registro de LOC
4	PM	02/02/17	17:17:18		17:21:11	3.9	Registro de tiempos en las formas

7. Forma de registro de defectos.

Project	Date	Num	Type	Injected	Removed	FixTime	Fix Ref.	Description
4	02/02/2017	13	20	CODE	COMPILE	1.8		Se corrigió la sintaxis en la declaración de variables del ciclo "for" en el método computeB1 for (let Xi = listX.begin(), let Yi = listY.begin(); => for (let Xi = listX.begin(), Yi = listY.begin();

8. Estándar de codificación.

Plantilla de Estándar de Codificación

Propósito	Guiar en el desarrollo de programas de software
Encabezado de programas	Comenzar todos los programas con un encabezado descriptivo.
Formato de encabezado	/********** /* Name: nombre de programador. */ /* Date: la fecha en la que se inició el desarrollo del programa. */ /* Description: una corta descripción del programa y lo que hace. */ **********
Instrucciones de reutilización	• Describir cómo es usado el programa. Proveer el formato de declaración, valores y tipos así como los límites de los parámetros. • Proveer advertencias de valores ilegales, condiciones de sobre flujo o cualquier otra condición que pudiera resultar en una operación impropia.
Ejemplo de reutilización	/********** /* Reuse instructions */ /* printLine(lineOfCharacter) */ /* Purpose: to print string, 'lineOfCharacter', on one print line */ /* Limitations: the line length must not exceed LINE_LENGTH */ /* Return 0 if printer not ready to print, else 1 */ **********
Identificadores	Usar nombres descriptivos para todas las variables, funciones, constantes y cualquier otro identificador. Evitar abreviaciones o el uso de una única letra.
Variables	• Se deberá usar el estilo de escritura "*lowerCamelCase*" para nombrar las variables. Ejemplos: • var lineOfCharacter = 'Hola mundo'; • var isEmpty = true; • var counter = 0;

Constantes	• Se deberá usar puras mayúsculas y separadas por "_" en caso de dos o más palabras para nombrar las constantes. Ejemplos: • const SIZE = 100; • const LINE_LENGTH = 1000;
Métodos/Funciones	• Se deberá usar verbos en infinitivo para el nombre de los métodos. • Se deberá usar el estilo de escritura "*lowerCamelCase*" Ejemplos: • run () {} • draw () {}

Clases	• Se deberá usar sustantivos para nombrar las clases. • Se deberá usar el estilo de escritura "UpperCamelCase" para nombrar las clases. Ejemplos: • class Math {} • class WoodenChair {}
Comentarios	• Documentar el código de tal manera que el lector pueda entender su operación. • Los comentarios deben explicar tanto el propósito y el comportamiento del código. • Comentar la declaración de variables para indicar su propósito. • Se deberá usar dos tipos de comentario corto y largo. Ejemplos: • Largo: /* Comentario largo … */ • Corto: // Comentario corto.
Ejemplos de buenos comentarios	// have all records been processed? if (recordCount > limit) {}
Ejemplo de malos comentarios	// check if record count exceeds limit if (recordCount > limit) {}
Secciones principales	Preceder las secciones por un bloque de comentarios que describa el tipo de procesamiento que hacen.
Ejemplo	/**/ /* The program section examines the contents of the array 'grades' and calcu- */ /* lates the average class grade. */ /**/
Espacios en blaco	• Escribir los programas con suficiente espacio en blanco para que sea legible. • Separa cada constructor de programa con al menos un espacio en blanco.
Identación	• Identar cada nivel o rama respecto a la anterior. • Cerrar y abrir los niveles o ramas en su propia línea alineándolas.

Ejemplo de identación	while (missDistance > threshold) {
	  successCode = moveRobot (targetLocation);
	  if (successCode === MOVE_FAILED) {
	  Console.log('The robot move has failed.');
	  }
	}

Uso de mayúsculas y minúsculas	• En los mensajes hacia los usuarios pueden usarse tanto minúsculas como mayúsculas para dejar en claro el contenido.

<table>
<tr>
<td>Declaraciones de variables</td>
<td>

Se deberá realizar una única declaración por línea e inicializarlas.
Se deberá declarar las variables al comienzo de los bloques.
Las variables locales se deberán declarar en el bloque donde se utilicen.

Ejemplos:
<pre>
let base = 100;
method()
{
  let count = 0;
  while(count < base) {
    console.log(base + count);
    count++;
  }
}
</pre>
</td>
</tr>
<tr>
<td>Funciones/Métodos</td>
<td>
<pre>
function nameFunction()
{
  ...
}
nameMethod ()
{
  ...
}
(param) => {
  ...
}
</pre>
</td>
</tr>
<tr>
<td>Declaraciones de clases</td>
<td>
<pre>
class List {
    constructor()
    {
        this.head = null;
        this.tail = null;
    }

    get isEmpty()
    {
        return this.head === null;
    }
}
</pre>
</td>
</tr>
</table>

Declaraciones de estructuras de control de flujo.	let isEmpty = true;
	let count = 0;
	
	if (isEmpty) {
	…
	}
	if (isEmpty) {
	…
	} else {
	…
	}
	if (isEmpty) {
	…
	} else if (count > 10) {
	…
	} else {
	…
	}
	
	let char = 'r';
	switch (char) {
	case 'r': …
	break;
	default: …
	break;
	}
	
	try {
	…
	} catch (e) {
	…
	}
	
	try {
	…
	} catch (e) {
	…
	} finally {
	…
	}

Declaraciones de estructuras de iteración	for (let i=0; i<10; i++) {
	…
	}
	
	for (let object in objects) {
	…
	}
	
	for (let value of values) {
	…
	}
	
	while (value < 10) {
	…
	value++;
	}
	
	do {
	…
	} while(value < 10);

9. Código fuente del programa.

```
/*************************************************************************/
/*  Name:        Rubén ignacio Couoh Ku.                                 */
/*  Date:        11/01/2017                                              */
/*  Description: Nodo para construcción de la lista ligada.              */
/*************************************************************************/

class Node {

    constructor(value, next)
    {
        this.value = value;
        this.next = next;
    }
}

module.exports = Node;
```

```
/*************************************************************************/
/*  Name:        Rubén ignacio Couoh Ku.                                 */
/*  Date:        11/01/2017                                              */
/*  Description: Lista ligada simple.                                    */
/*************************************************************************/

var Node = require('./Node.js');

class List {

    constructor(values=[])
    {
        this.head = null;
        this.tail = null;
        this._size = 0;

        for (let value of values) {
            this.push(value);
        }
    }

    begin ()
    {
        let first = this.head;
        return first;
    }

    end ()
    {
        let last = this.tail;
        return last;
    }

    get isEmpty()
    {
        return this._size === 0;
    }
```

```
38
39      push(value)
40      {
41          let newNode = new Node(value, null);
42
43          if (this.isEmpty) {
44              this.head = newNode;
45              this.tail = this.head;
46          } else {
47              this.tail.next = newNode;
48              this.tail = this.tail.next;
49          }
50
51          this._size = this._size + 1;
52      }
53
54      pop()
55      {
56          let value = null;
57          if (!this.isEmpty) {
58              let newFirst = this.head.next;
59              value = this.head.value;
60
61              if (this.head === this.tail) {
62                  delete this.head;
63                  this.head = this.tail = null;
64              } else {
65                  delete this.head;
66                  this.head = this.newFirst;
67              }
68
69              this._size = this._size - 1;
70          }
71
72          return value;
73      }
74
```

```
75      peek()
76      {
77          let value = null;
78          if (!this.isEmpty) {
79              value = this.head.value;
80          }
81
82          return value;
83      }
84
85
86      get size()
87      {
88          return this._size;
89      }
90  }
91
92  module.exports = List;
```

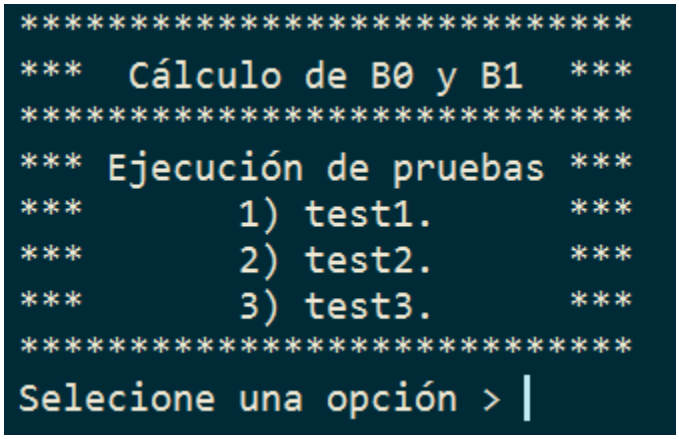
```
1  /*************************************************************************/
2  /*  Name:        Rubén ignacio Couoh Ku.                                 */
3  /*  Date:        12/01/2017                                              */
4  /*  Description: Calcula la media y la desviación standar de una lista   */
5  /*************************************************************************/
6
7  class RMath {
8
9      constructor()
10     {
11     }
12
13
14     /*****************************************************************************************/
15     /*  Reuse Instructions                                                                   */
16     /*      toAverage(list)                                                                  */
17     /*      Purpose:     Calcula la media de un conjunto de datos recibidos de list.         */
18     /*      Limitations: NA                                                                  */
19     /*      Return:      Regresa un numero real.                                             */
20     /*****************************************************************************************/
21     static toAverage(list)
22     {
23         let node = list.begin();
24         let mean = 0;
25         let sum  = 0;
26
27         if (!list.isEmpty) {
28             while ( node !== null) {
29                 sum += node.value;
30                 node = node.next;
31             }
32         }
33         mean = sum / list.size;
34         return mean;
35     }
```

```
37      /*****************************************************************************************/
38      /*  Reuse Instructions                                                                   */
39      /*      computeB1(listX, listY)                                                          */
40      /*      Purpose:     Calcula el parámetro B1 de la regresión lineal listX y listY.       */
41      /*      Limitations: NA                                                                  */
42      /*      Return:      Regresa un numero real.                                             */
43      /*****************************************************************************************/
44      static computeB1(listX, listY)
45      {
46          let n = listX.size;
47          let Xavg = RMath.toAverage(listX);
48          let Yavg = RMath.toAverage(listY);
49          let B1  = 0;
50          let sumXiYi = 0;
51          let sumXiXi = 0;
52
53          for (let Xi = listX.begin(), Yi = listY.begin(); Xi !== null; Xi = Xi.next, Yi = Yi.next) {
54              sumXiYi += Xi.value * Yi.value;
55              sumXiXi += Math.pow(Xi.value, 2);
56          }
57
58          B1 = (sumXiYi - n * Xavg * Yavg) / (sumXiXi - n * Math.pow(Xavg, 2));
59
60          return B1;
61      }
62
63      /*****************************************************************************************/
64      /*  Reuse Instructions                                                                   */
65      /*      computeB0(listX, listY, B1)                                                      */
66      /*      Purpose:     Calcula el parámetro B0 de la regresión lineal listX y listY y B1.  */
67      /*      Limitations: NA                                                                  */
68      /*      Return:      Regresa un numero real.                                             */
69      /*****************************************************************************************/
70      static computeB0(listX, listY, B1)
71      {
72          let Xavg = RMath.toAverage(listX);
73          let Yavg = RMath.toAverage(listY);
74          let B0 = 0;
75
76          B0 = Yavg - B1 * Xavg;
77
78          return B0;
79      }
80  }
81
82  module.exports = RMath;
```

```
/*************************************************************************/
/*  Name:        Rubén ignacio Couoh Ku.                                 */
/*  Date:        11/01/2017                                              */
/*  Description: Calcula la media y la desviación standar                */
/*************************************************************************/

const readline = require('readline');
const fs = require('fs');

const List = require('./List.js');
const RMath = require('./RMath.js');

class Main {

    static main()
    {
        let io = readline.createInterface({
            input: process.stdin,
            output: process.stdout
        });

        console.log('  ****************************');
        console.log('  ***  Cálculo de B0 y B1  ***');
        console.log('  ****************************');
        console.log('  *** Ejecución de pruebas ***');
        console.log('  ***       1) test1.      ***');
        console.log('  ***       2) test2.      ***');
        console.log('  ***       3) test3.      ***');
        console.log('  ****************************');

        io.question('  Selecione una opción > ', onInput);

        function onInput(input)
        {
            let file = null;

            input = Number.parseInt(input);
            switch (input) {
                case 1:
                case 2:
                case 3:
                    file = `test${input}.txt`;
                    readFile(file, onValues);
                break;
                default:
                    console.log('¡Opción incorrecta!');
```

```
48                  break;
49              }
50              io.close();
51          }
52
53          function readFile(file, cb)
54          {
55              fs.readFile(file, 'utf-8', (err, content) => {
56                  let data = { X: [], Y: []};
57                  let lines = [];
58
59                  if (!err) {
60
61                      lines = content.split('\r\n');
62                      data = lines.reduce((result, line) => {
63                          let pair = line.split(' ');
64                          result.X.push(Number.parseFloat(pair[0]));
65                          result.Y.push(Number.parseFloat(pair[1]));
66                          return result;
67                      }, data);
68                  }
69
70                  cb(err, data);
71              });
72          }
73
74          function onValues(err, data)
75          {
76              if (err) {
77                  process.exit(0);
78              }
79
80              var listX = new List(data.X);
81              var listY = new List(data.Y);
82
83              let B1 = RMath.computeB1(listX, listY);
84              let B0 = RMath.computeB0(listX, listY, B1);
85
86              console.log(`B0: ${B0.toFixed(2)}`);
87              console.log(`B1: ${B1.toFixed(4)}`);
88          }
89      }
90
91  }
92
93  Main.main();
```

10. Pantallas de la interfaz gráfica.



11. Pantalla de resultados.

Entrada test1.txt	Salida
130 186 650 699 99 132 150 272 128 291 302 331 95 199 945 1890 368 788 961 1601	***************************** *** Cálculo de B0 y B1 *** ***************************** *** Ejecución de pruebas *** *** 1) test1. *** *** 2) test2. *** *** 3) test3. *** ***************************** Selecione una opción > 1 B0: -22.55 B1: 1.7279

Entrada test2.txt	Salida
163 186 765 699 141 132 166 272 137 291 355 331 136 199 1206 1890 433 788 1130 1601	******************************** *** Cálculo de B0 y B1 *** ******************************** *** Ejecución de pruebas *** *** 1) test1. *** *** 2) test2. *** *** 3) test3. *** ******************************** Selecione una opción > 2 B0: -23.92 B1: 1.4310

Entrada test3.txt	Salida
80 83 65 60 57 59	******************************** *** Cálculo de B0 y B1 *** ******************************** *** Ejecución de pruebas *** *** 1) test1. *** *** 2) test2. *** *** 3) test3. *** ******************************** Selecione una opción > 3 B0: -7.16 B1: 1.1064

12. Tabla de resultados:

Prueba	Resultados esperados		Resultados actuales	
	β0	β1	β0	β1
Tabla: Objetos estimados vs. LOC nuevas y cambiadas actuales	-22.55	1.7279	-22.55	1.7279
Tabla. LOC nuevas y cambiadas estimadas vs. LOC nuevas y cambiadas actuales	-23.92	1.4310	-23.92	1.4310
Programa 2A, 3A, 4A LOC nuevas y cambiadas estimadas vs. LOC nuevas y cambiadas actuales	-7.16	1.1064	-716	1.1064