

Informe de Laboratorio 02

Tema: Arreglos estándar

Nota

Estudiante	Escuela	Asignatura
Christian Mestas Zegarra cmestasz@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la Programación 2 Semestre: II Código: 1701213

Laboratorio	Tema	Duración
02	Arreglos estándar	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 13 Setiembre 2023	Al 20 Setiembre 2023

1. Tarea

- **Actividad 1:** En este ejercicio se le solicita a usted implementar el juego del ahorcado utilizando el código parcial que se le entrega. Deberá considerar que:
 - El juego valida el ingreso de letras solamente. En caso el usuario ingrese un carácter equivocado le dará el mensaje de error y volverá a solicitar el ingreso.
 - El juego supone que el usuario no ingresa una letra ingresada previamente.
 - El método `ingreseLetra()` debe ser modificado para incluir las consideraciones de validación.
 - Puede crear métodos adicionales.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Microsoft Windows 10 Pro 64 bits
- Visual Studio Code 1.82.2
- Java Development Kit 17.0.1
- Git 2.41.0.windows.1
- Windows PowerShell 5.1.19041.3031
- Cuenta en GitHub con el correo institucional.
- Arreglos estándar.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/cmestasz/fp2-23b.git`
- URL para el laboratorio 01 en el Repositorio GitHub.
- `https://github.com/cmestasz/fp2-23b/tree/main/fase01/lab01`

4. Actividades con el repositorio GitHub

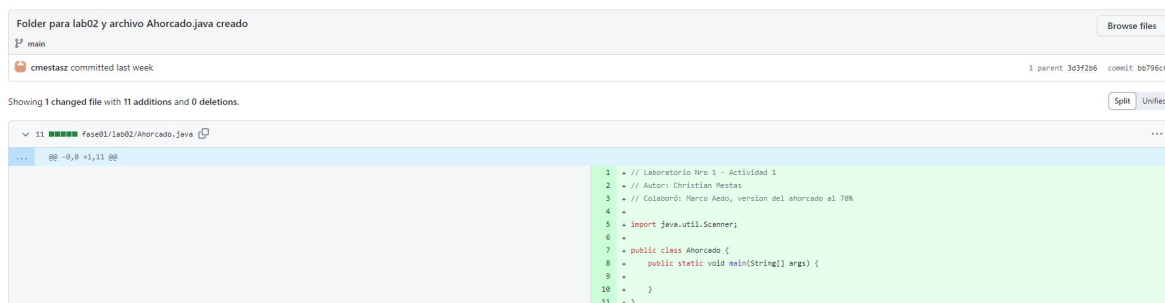
Creando Ahorcado.java

```
$ mkdir lab02  
$ cd lab02  
$ code Ahorcado.java
```

Primer Commit / Ahorcado.java

```
$ git add Ahorcado.java  
$ git commit -m "Folder para lab02 y archivo Ahorcado.java creado"  
$ git push
```

Commit



Primer Commit.

Actualizando Ahorcado.java

```
$ code Ahorcado.java
```

Segundo - Septimo Commit / Ahorcado.java

```
$ git add Ahorcado.java
$ git commit -m "Actividad 1"
$ code Ahorcado.java
$ git add Ahorcado.java
$ git commit -m "Actividad 2"
$ code Ahorcado.java
$ git add Ahorcado.java
$ git commit -m "Actividad 3"
$ code Ahorcado.java
$ git add Ahorcado.java
$ git commit -m "Actividad 4"
$ code Ahorcado.java
$ git add Ahorcado.java
$ git commit -m "Actividad 5"
$ git push
```

Commit

Main con las variables y metodos que se planean usar y crear, ademas ...
...de un ciclo de juego basico

17 main

emestaz committed last week

Showing 1 changed file with 69 additions and 0 deletions.

Assets/12002/Ahorcado.java

```

6  public class Ahorcado {
7      public static void main(String[] args) {
8          // ...
9          String ahorc1 = "-----";
10         // ...
11         // ...
12         // ...
13         // ...
14         // ...
15         // ...
16         String ahorc2 = "-----";
17         // ...
18         // ...
19         // ...
20         // ...
21         // ...
22         // ...
23         String ahorc3 = "-----";
24         // ...
25         // ...
26         // ...
27         // ...
28         // ...
29         // ...
30         String ahorc4 = "-----";
31         // ...
32         // ...
33         // ...
34         // ...
35         // ...
36         // ...
37         String ahorc5 = "-----";
38         // ...
39         // ...
40         // ...
41         // ...
42         // ...
43         // ...
44         String ahorc6 = "-----";
45         // ...
46         // ...

```

Segundo Commit.

Commit

Metodos parcialmente ya implementados, con espacios para completar

main

cmestasz committed last week

Showing 1 changed file with 34 additions and 0 deletions.

```

1  @@ -77,4 +77,38 @@ public static void main(String[] args) {
2      77
3      78     System.out.println("a");
4      79 }
5
6      80 +
7      81 +     public static String getPalabrasSecreta(String[] palabras) {
8      82 +         String palabraSecreta;
9      83 +         int ind;
10     84 +         int indiceMayor = palabras.length - 1;
11     85 +         int indiceMenor = 0;
12     86 +         ind = (int) (Math.random() * (indiceMayor - indiceMenor + 1) + indiceMenor);
13     87 +         return palabras[ind];
14     88 +     }
15     89 +
16     90 +     public static void mostrarBlancos(String palabra) {
17     91 +         for (int i = 0; i < palabra.length(); i++)
18     92 +             System.out.print(" ");
19     93 +     }
20     94 +
21     95 +     public static String ingresarLetra() {
22     96 +         String letra;
23     97 +         Scanner sc = new Scanner(System.in);
24     98 +         System.out.print("Ingrese letra: ");
25     99 +         letra = sc.next();
26    100 +         while (letra.length() != 1) {
27    101 +             System.out.print("Ingrese letra: ");
28    102 +             letra = sc.next();
29    103 +         }
30    104 +         return letra;
31    105 +     }
32    106 +
33    107 +     public static boolean letraEnPalabrasSecreta(String letra, String palabras) {
34    108 +         return false;
35    109 +     }
36    110 +
37    111 +     public static void mostrarBlancosActualizados(String letra) {
38    112 +         System.out.print("PROCESANDO....");
39    113 +     }
40    114 + }

```

Tercer Commit.

Commit

Metodo letraEnPalabrasSecreta

main

cmestasz committed last week

Showing 1 changed file with 11 additions and 4 deletions.

```

1  @@ -1,4 +1,4 @@
2  1 // Laboratorio 1 - Actividad 1
3  2 // Autor: Christian Mestas
4  3 // Colaborador: Marco Andujar, versión del shorcado al 70%
5  4
6  5 @@ -74,7 +74,8 @@ public static void main(String[] args) {
7  74     System.out.println("Figuras: " + contador);
8  75     contador = contador + 1;
9  76 }
10 77 +
11 78 + //7000: Indicar al usuario, perdida y cantidad de turnos
12 79 +
13 80 + System.out.println("a");
14 81 }
15 82
16 83 @@ -89,11 +89,11 @@ public static String getPalabrasSecreta(String[] palabras) {
17 89
18 90     public static void mostrarBlancos(String palabra) {
19 91         for (int i = 0; i < palabra.length(); i++)
20 92             System.out.print(" ");
21 93     }
22 94
23 95     public static String ingresarLetra() {
24 96         String letra;
25 97         Scanner sc = new Scanner(System.in);
26 98         System.out.print("Ingrese letra: ");
27 99         letra = sc.next();
28 100         while (letra.length() != 1) {
29 101             System.out.print("Ingrese letra: ");
30 102             letra = sc.next();
31 103         }
32 104         return letra;
33 105     }
34 106
35 107     public static boolean letraEnPalabrasSecreta(String letra, String palabras) {
36 108         char caracter = letra.charAt(0);
37 109         for (int i = 0; i < palabras.length(); i++) {
38 110             if (palabras.charAt(i) == caracter)
39 111                 return true;
40 112         }
41 113         return false;
42 114     }
43 115
44 116     public static void mostrarBlancosActualizados(String letra) {
45 117         //7000: metodo incompleto

```

Cuarto Commit.

Commit

Metodo esLetraValida para validar entrada en metodo ingreseLetra

main

cmestasz committed last week

1 parent d315a88 commit 6d7b714

Showing 1 changed file with 9 additions and 2 deletions.

Split Unified

```

11 fased01/lab02/Ahorcado.java
@@ -96,9 +96,9 @@ public static void mostrarBlancos(String palabra) {
96 public static String ingreseLetra() {
97 String laLetra;
98 Scanner sc = new Scanner(System.in);
99 - System.out.println("Ingrese letra: "); //TODO: validar caracteres permitidos
100 laLetra = sc.next();
101 - while (laLetra.length() != 1) {
102 System.out.println("Ingrese letra: ");
103 laLetra = sc.next();
104 }
118 //TODO: metodo incompleto
119 System.out.println("PROCESANDO....");
120 }
121 }

96 public static String ingreseLetra() {
97 String laLetra;
98 Scanner sc = new Scanner(System.in);
99 + System.out.println("Ingrese letra: ");
100 laLetra = sc.next();
101 + while (!esLetraValida(laLetra)) {
102 System.out.println("Ingrese letra: ");
103 laLetra = sc.next();
104 }
122 +
123 + public static boolean esLetraValida(String letra) {
124 + if (letra.length() != 1)
125 + return false;
126 + char caracter = letra.charAt(0);
127 + return Character.isLetter(caracter);
128 + }

```

Quinto Commit.

Commit

Metodo mostrarBlancosActualizados, con una manera de llevar la cuenta...

de las letras encontradas

main

cmestasz committed last week

1 parent 6d7b714 commit bd59911

Showing 1 changed file with 16 additions and 6 deletions.

Split Unified

```

22 fased01/lab02/Ahorcado.java
@@ -66,10 +66,11 @@ public static void main(String[] args) {
66 mostrarBlancos(palSecreta);
67 System.out.println("W");
68
69 while (contador <= 0) {
70 letra = ingreseLetra();
71 if (letraValida(letra, palSecreta)) {
72 - mostrarBlancosActualizados(letra);
73 + mostrarBlancosActualizados(letra, palSecreta, letrasEncontradas);
74 }
75 System.out.println("Figuras: " + contador);
76 contador = contador + 1;
77 }
187 }
188
189 public static boolean letraValida(String letra, String palSecreta) {
190 char letraValida = letra.charAt(0);
191 for (int i = 0; i < palSecreta.length(); i++) {
192 if (palSecreta.charAt(i) == letraValida) {
193 return true;
194 }
195 return false;
196 }
197 }
198
199 public static void mostrarBlancosActualizados(String letra) {
200 //TODO: metodo incompleto
201 System.out.println("PROCESANDO....");
202 }
203
204 public static boolean esLetraValida(String letra) {
205 }
206
207 public static void mostrarBlancosActualizados(String letra, String palSecreta, boolean[] letrasEncontradas) {
208 char letraValida = letra.charAt(0);
209 for (int i = 0; i < palSecreta.length(); i++) {
210 char letraActual = palSecreta.charAt(i);
211 if (letraActual == letraValida && !letrasEncontradas[i]) {
212 System.out.println(letraActual + " ");
213 letrasEncontradas[i] = true;
214 } else {
215 System.out.print("_ ");
216 }
217 }
218 System.out.println("\n");
219 }
220
221 public static boolean esLetraValida(String letra) {
222 }

```

Sexto Commit.

Commit

Metodo verificarPalabraCompleta para completar Main (indicar si gano ...
-> perdis y cuantos turnos duro el juego)

main
cmestasz committed last week
1 parent: 9d539952 commit: 7db72d5

Showing 1 changed file with 22 additions and 6 deletions.

Files

File	Changes
src/main/java/VerificarPalabraCompleta.java	22 additions, 6 deletions

```

57  "=====
58
59  String[] figuras = {ahor1, ahor2, ahor3, ahor4, ahor5, ahor6, ahor7};
60  int contador = 1;
61  String letra;
62  String[] palabras = {"programacion", "java", "indentacion", "clases", "objetos", "desarrollador", "pruebas"};
63
64  @Override @Override public void main(String[] args) {
65      System.out.println("u");
66      boolean[] letrasContraidas = new boolean[palabras.length()];
67      while (contador <= 6) {
68          letra = ingresarLetra();
69          if (letrasContraidas[letra, palabras]) {
70              mostrarLetrasContraidas(letra, palabras, letrasContraidas);
71          } else {
72              System.out.println(figuras[contador]);
73              contador = contador + 1;
74          }
75      }
76      //TODO: indicar si gano, perdis y cantidad de turnos
77
78
79      System.out.println("u");
80  }
81
82  @Override @Override public static boolean esLetraValida(String letra) {
83      char caracter = letra.charAt(0);
84      return Character.isLetter(caracter);
85  }
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

Séptimo Commit.

5. Código desarrollado

Ahorcado.java

```
1 // Laboratorio Nro 2 - Actividad 1
2 // Autor: Christian Mestas
3 // Colaboro: Marco Aedo, version del ahorcado al 70%
4
5 import java.util.Scanner;
6
7 public class Ahorcado {
8     public static void main(String[] args) {
9         String ahor1 = " +---+ \n" +
10             " | | \n" +
11             " | \n" +
12             " | \n" +
13             " | \n" +
14             " | \n" +
15             "===== ";
16         String ahor2 = " +---+ \n" +
17             " | | \n" +
18             " 0 | \n" +
19             " | \n" +
20             " | \n" +
21             " | \n" +
22             "===== ";
23         String ahor3 = " +---+ \n" +
24             " | | \n" +
25             " 0 | \n" +
26             " | | \n" +
27             " | \n" +
28             " | \n" +
29             "===== ";
30         String ahor4 = " +---+ \n" +
31             " | | \n" +
32             " 0 | \n" +
33             " /| | \n" +
34             " | \n" +
35             " | \n" +
36             "===== ";
37         String ahor5 = " +---+ \n" +
38             " | | \n" +
39             " 0 | \n" +
40             " /|\| | \n" +
41             " | \n" +
42             " | \n" +
43             "===== ";
44         String ahor6 = " +---+ \n" +
45             " | | \n" +
46             " 0 | \n" +
47             " /|\| | \n" +
48             " / | \n" +
49             " | \n" +
50             "===== ";
51         String ahor7 = " +---+ \n" +
52             " | | \n" +
```

```
53         " 0 | \n" +
54         "/|\\ | \n" +
55         "/ \\ | \n" +
56         " | \n" +
57         "=====";
58
59     String[] figuras = {ahor1, ahor2, ahor3, ahor4, ahor5, ahor6, ahor7};
60     int contador = 1, turnos = 0;
61     String letra;
62     String[] palabras = {"programacion", "java", "indentacion", "clases", "objetos",
        "desarrollador", "pruebas"};
63
64     String palSecreta = getPalabraSecreta(palabras);
65     System.out.println(figuras[0]);
66     mostrarBlancos(palSecreta);
67     System.out.println("\n");
68
69     boolean[] letrasEncontradas = new boolean[palSecreta.length()];
70     while (contador <= 6 && !verificarPalabraCompleta(letrasEncontradas)) {
71         letra = ingreseLetra();
72         if (letraEnPalabraSecreta(letra, palSecreta)) {
73             mostrarBlancosActualizados(letra, palSecreta, letrasEncontradas);
74         }
75         else {
76             System.out.println(figuras[contador]);
77             contador = contador + 1;
78         }
79         turnos = turnos + 1;
80     }
81
82     if (verificarPalabraCompleta(letrasEncontradas))
83         System.out.println("Has ganado!");
84     else
85         System.out.println("Has perdido!");
86     System.out.println("El juego duro " + turnos + " turnos.");
87
88     System.out.println("\n");
89 }
90
91 public static String getPalabraSecreta(String[] lasPalabras) {
92     String palSecreta;
93     int ind;
94     int indiceMayor = lasPalabras.length - 1;
95     int indiceMenor = 0;
96     ind = (int) ((Math.random() * (indiceMayor - indiceMenor + 1) + indiceMenor));
97     return lasPalabras[ind];
98 }
99
100 public static void mostrarBlancos(String palabra) {
101     for (int i = 0; i < palabra.length(); i++)
102         System.out.print("_ ");
103 }
104
105 public static String ingreseLetra() {
106     String laLetra;
107     Scanner sc = new Scanner(System.in);
```



```
108     System.out.println("Ingrese letra: ");
109     laLetra = sc.next();
110     while (!esLetraValida(laLetra)) {
111         System.out.println("Ingrese letra: ");
112         laLetra = sc.next();
113     }
114     return laLetra;
115 }
116
117 public static boolean letraEnPalabraSecreta(String letra, String palSecreta) {
118     char letraIngresada = letra.charAt(0);
119     for (int i = 0; i < palSecreta.length(); i++) {
120         if (palSecreta.charAt(i) == letraIngresada)
121             return true;
122     }
123     return false;
124 }
125
126 public static void mostrarBlancosActualizados(String letra, String palSecreta, boolean[]
127     letrasEncontradas) {
128     char letraIngresada = letra.charAt(0);
129     for (int i = 0; i < palSecreta.length(); i++) {
130         char letraActual = palSecreta.charAt(i);
131         if (letraActual == letraIngresada || letrasEncontradas[i]) {
132             System.out.print(letraActual + " ");
133             letrasEncontradas[i] = true;
134         } else {
135             System.out.print("_ ");
136         }
137     }
138     System.out.println("\n");
139 }
140
141 public static boolean esLetraValida(String letra) {
142     if (letra.length() != 1)
143         return false;
144     char caracter = letra.charAt(0);
145     return Character.isLetter(caracter);
146 }
147
148 public static boolean verificarPalabraCompleta(boolean[] letrasEncontradas) {
149     for (int i = 0; i < letrasEncontradas.length; i++) {
150         if (!letrasEncontradas[i])
151             return false;
152     }
153     return true;
154 }
```

- Se inicializan variables con los posibles estados del juego, posibles palabras secretas y un contador de turnos.
- `getPalabraSecreta()` regresa una palabra escogida al azar del arreglo palabras.
- `mostrarBlancos()` imprime la cantidad de letras que se tienen que descubrir.

7. Estructura de laboratorio 02

- El contenido que se entrega en este laboratorio es el siguiente:

```
lab01/  
|--- Ahorcado.java  
|--- Informe.tex  
|--- Informe.pdf  
|--- img  
|   |--- logo_abet.png  
|   |--- logo_episunsa.png  
|   |--- logo_unsa.jpg  
|   |--- commit01.jpg  
|   |--- commit02.jpg  
|   |--- commit03.jpg  
|   |--- commit04.jpg  
|   |--- commit05.jpg  
|   |--- commit06.jpg  
|   |--- commit07.jpg  
|   |--- ejec01.jpg
```

8. Rúbricas

8.1. Entregable Informe

Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y fácil de leer.

8.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumple con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobatoria, siempre y cuando cumpla con todos los ítems.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Niveles de desempeño

Puntos	Nivel			
	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	1.5	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	1.5	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2		0	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		17	

9. Referencias

- Aedo, M. y Castro, E. (2021). FUNDAMENTOS DE PROGRAMACIÓN 2 - Tópicos de Programación Orientada a Objetos. Editorial UNSA.