

# PROGRAMACIÓN II

## Trabajo Práctico 2: Programacion Estructurada

### 1. Verificación de Año Bisiesto.

Escribe un programa en Java que solicite al usuario un año y determine si es bisiesto. Un año es bisiesto si es divisible por 4, pero no por 100, salvo que sea divisible por 400.

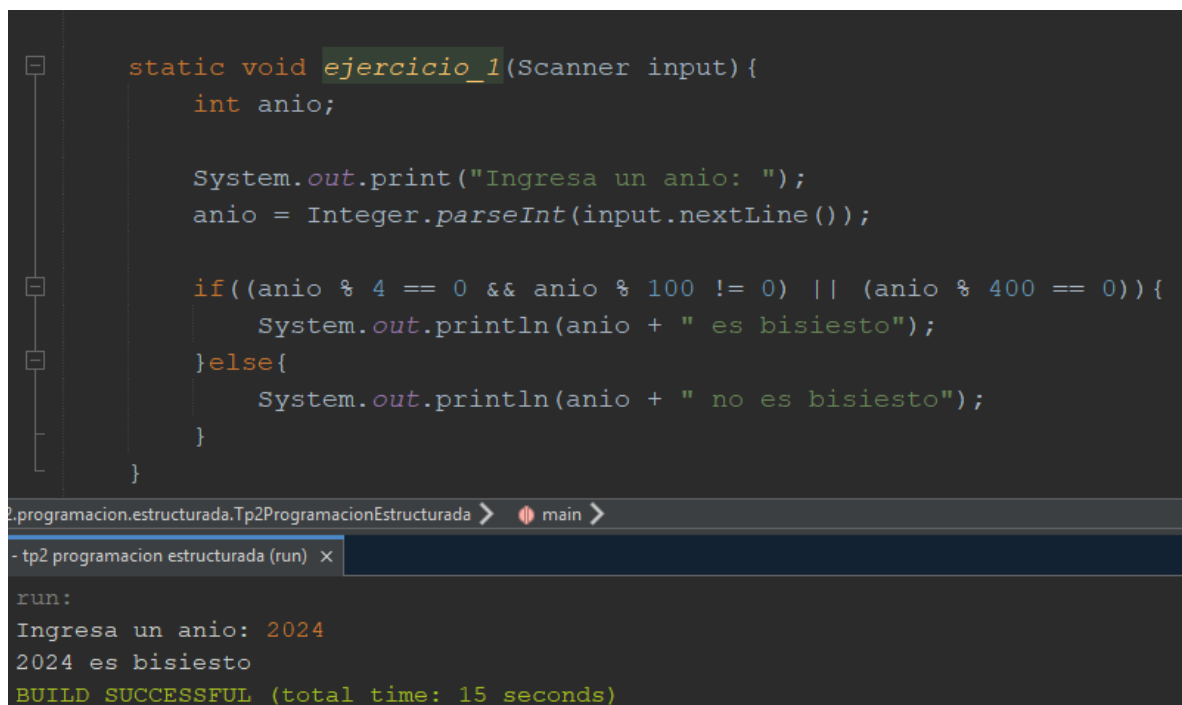
Ejemplo de entrada/salida:

Ingrese un año: 2024

El año 2024 es bisiesto.

Ingrese un año: 1900

El año 1900 no es bisiesto.



```
static void ejercicio_1(Scanner input){
    int anio;

    System.out.print("Ingresa un anio: ");
    anio = Integer.parseInt(input.nextLine());

    if((anio % 4 == 0 && anio % 100 != 0) || (anio % 400 == 0)){
        System.out.println(anio + " es bisiesto");
    }else{
        System.out.println(anio + " no es bisiesto");
    }
}

2.programacion.estructurada.Tp2ProgramacionEstructurada > main >
- tp2 programacion estructurada (run) x
run:
Ingresa un anio: 2024
2024 es bisiesto
BUILD SUCCESSFUL (total time: 15 seconds)
```

### 2. Determinar el Mayor de Tres Números.

Escribe un programa en Java que pida al usuario tres números enteros y

determine cuál es el mayor.

Ejemplo de entrada/salida:

Ingrese el primer número: 8

Ingrese el segundo número: 12

Ingrese el tercer número: 5

El mayor es: 12

```
39  static void ejercicio_2(Scanner input){
40      int num1, num2, num3, mayor;
41
42      System.out.print("Ingresa el primer numero: ");
43      num1 = Integer.parseInt(input.nextLine());
44      System.out.print("Ingresa el segundo numero: ");
45      num2 = Integer.parseInt(input.nextLine());
46      System.out.print("Ingresa el tercer numero: ");
47      num3 = Integer.parseInt(input.nextLine());
48
49      mayor = num1;
50
51      if(num2 > mayor) {
52          mayor = num2;
53      }if(num3 > mayor) {
54          mayor = num3;
55      }
56      System.out.println("El mayor es: " + mayor);
57  }
58
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >

Output - tp2 programacion estructurada (run) x

run:

Ingresa el primer numero: 8

Ingresa el segundo numero: 12

Ingresa el tercer numero: 5

El mayor es: 12

BUILD SUCCESSFUL (total time: 11 seconds)

### 3. Clasificación de Edad.

Escribe un programa en Java que solicite al usuario su edad y clasifique su etapa de vida según la siguiente tabla:

Menor de 12 años: "Niño"

Entre 12 y 17 años: "Adolescente"

Entre 18 y 59 años: "Adulto"

60 años o más: "Adulto mayor"

Ejemplo de entrada/salida:

Ingrese su edad: 25

Eres un Adulto.

Ingrese su edad: 10

Eres un Niño.

```
58
59 static void ejercicio_3(Scanner input){
60     int edad;
61
62     System.out.print("Ingresa tu edad: ");
63     edad = Integer.parseInt(input.nextLine());
64
65     if(edad < 12) {
66         System.out.println("Niño");
67     }else if(edad >= 12 && edad <= 17 ) {
68         System.out.println("Adolescente");
69     }else if(edad >= 18 && edad <= 59 ) {
70         System.out.println("Adulto");
71     }else{
72         System.out.println("Adulto mayor");
73     }
74
75 }
76
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >

tp2 - tp2 programacion estructurada (run) x

```
run:
Ingresa tu edad: 25
Adulto
BUILD SUCCESSFUL (total time: 5 seconds)
```

#### 4. Calculadora de Descuento según categoría.

Escribe un programa que solicite al usuario el precio de un producto y su categoría (A, B o C).

Luego, aplique los siguientes descuentos:

Categoría A: 10% de descuento

Categoría B: 15% de descuento

Categoría C: 20% de descuento

El programa debe mostrar el precio original, el descuento aplicado y el precio final

Ejemplo de entrada/salida:

Ingrese el precio del producto: 1000

Ingrese la categoría del producto (A, B o C): B

Descuento aplicado: 15%

Precio final: 850.0

```
77 static void ejercicio_4(Scanner input) {
78     double precio, precioFinal;
79     String categoria;
80
81     System.out.print("Ingrese precio del producto: ");
82     precio = Integer.parseInt(input.nextLine());
83     System.out.print("Ingrese la categoria del producto (Selecciona: A, B o C): ");
84     categoria = input.nextLine();
85
86     if(categoria.equalsIgnoreCase("A")) {
87         precioFinal = precio - (precio * 0.10);
88         System.out.println("Descuento aplicado: 10%");
89     } else if(categoria.equalsIgnoreCase("B")) {
90         precioFinal = precio - (precio * 0.15);
91         System.out.println("Descuento aplicado: 15%");
92     } else if(categoria.equalsIgnoreCase("C")) {
93         precioFinal = precio - (precio * 0.20);
94         System.out.println("Descuento aplicado: 20%");
95     } else {
96         System.out.println("Ingresa una de las tres opciones (A, B o C)");
97         return;
98     }
99     System.out.println("Precio final: " + precioFinal);
100 }
101
```

tp2\_programacion.estructurada.Tp2ProgramacionEstructurada > main >

Output - tp2 programacion estructurada (run) x

```
run:
>> Ingrese precio del producto: 1000
>> Ingrese la categoria del producto (Selecciona: A, B o C): b
>> Descuento aplicado: 15%
>> Precio final: 850.0
BUILD SUCCESSFUL (total time: 10 seconds)
```

## Estructuras de Repetición:

### 5. Suma de Números Pares (while).

Escribe un programa que solicite números al usuario y sume solo los números pares. El ciclo debe continuar hasta que el usuario ingrese el número 0, momento en el que se debe mostrar la suma total de los pares ingresados.

Ejemplo de entrada/salida:

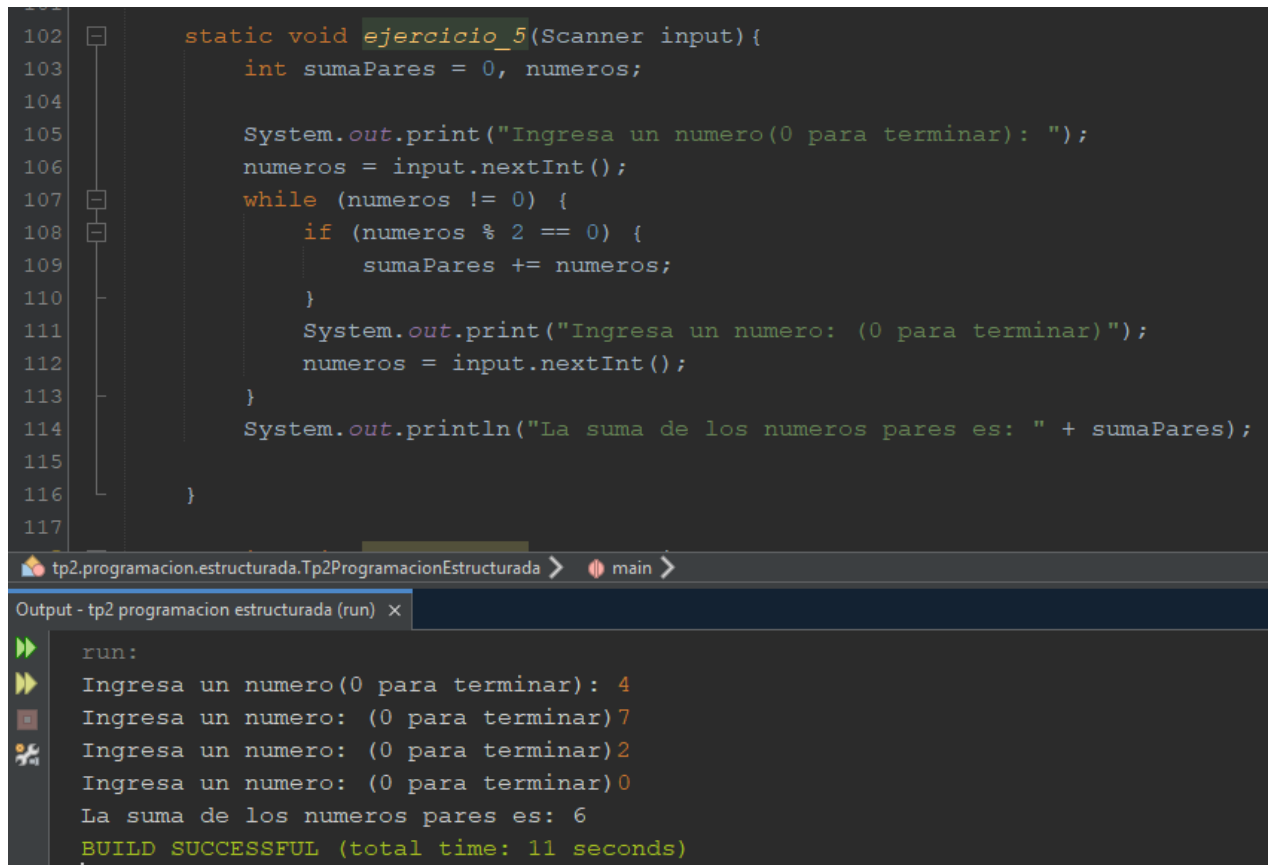
Ingrese un número (0 para terminar): 4

Ingrese un número (0 para terminar): 7

Ingrese un número (0 para terminar): 2

Ingrese un número (0 para terminar): 0

La suma de los números pares es: 6



```
102 static void ejercicio_5(Scanner input){
103     int sumaPares = 0, numeros;
104
105     System.out.print("Ingresa un numero(0 para terminar): ");
106     numeros = input.nextInt();
107     while (numeros != 0) {
108         if (numeros % 2 == 0) {
109             sumaPares += numeros;
110         }
111         System.out.print("Ingresa un numero: (0 para terminar)");
112         numeros = input.nextInt();
113     }
114     System.out.println("La suma de los numeros pares es: " + sumaPares);
115 }
116
117
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >

Output - tp2 programacion estructurada (run) x

```
run:
Ingresa un numero(0 para terminar): 4
Ingresa un numero: (0 para terminar)7
Ingresa un numero: (0 para terminar)2
Ingresa un numero: (0 para terminar)0
La suma de los numeros pares es: 6
BUILD SUCCESSFUL (total time: 11 seconds)
```

6. Contador de Positivos, Negativos y Ceros (for). Escribe un programa que pida al usuario ingresar 10 números enteros y cuente cuántos son positivos, negativos y cuántos son ceros.

Ejemplo de entrada/salida: Ingrese el número 1: -5 Ingrese el número 2: 3 Ingrese el número 3: 0 Ingrese el número 4: -1 Ingrese el número 5: 6 Ingrese el número 6: 0 Ingrese el número 7: 9 Ingrese el número 8: -3 Ingrese el número 9: 4 Ingrese el número 10: -8 Resultados: Positivos: 4 Negativos: 4 Ceros: 2

```
118 static void ejercicio_6(Scanner input){
119     int positivos = 0, negativos = 0, ceros = 0;
120
121     for (int i = 1; i <= 10; i++) {
122         System.out.print("Ingresa el numero " + i + " : ");
123         int numero = input.nextInt();
124
125         if(numero > 0){
126             positivos++;
127         }else if(numero < 0){
128             negativos++;
129         }else{
130             ceros++;
131         }
132     }
133     System.out.println("Resultados: ");
134     System.out.println("Positivos: " + positivos);
135     System.out.println("Negativos: " + negativos);
136     System.out.println("Ceros: " + ceros);
137 }
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >

Output - tp2 programacion estructurada (run) x

```
run:
Ingresa el numero 1 : -5
Ingresa el numero 2 : 3
Ingresa el numero 3 : 0
Ingresa el numero 4 : -1
Ingresa el numero 5 : 6
Ingresa el numero 6 : 0
Ingresa el numero 7 : 9
Ingresa el numero 8 : -3
Ingresa el numero 9 : 4
Ingresa el numero 10 : -8
Resultados:
Positivos: 4
Negativos: 4
Ceros: 2
BUILD SUCCESSFUL (total time: 23 seconds)
```

## 7. Validación de Nota entre 0 y 10 (do-while).

Escribe un programa que solicite al usuario una nota entre 0 y 10. Si el usuario ingresa un número fuera de este rango, debe seguir pidiéndole la nota hasta que ingrese un valor válido.

Ejemplo de entrada/salida:

Ingrese una nota (0-10): 15

Error: Nota inválida. Ingrese una nota entre 0 y 10.

Ingrese una nota (0-10): -2

Error: Nota inválida. Ingrese una nota entre 0 y 10.

Ingrese una nota (0-10): 8

Nota guardada correctamente.

```
139 static void ejercicio_7(Scanner input) {
140     int nota;
141
142     do {
143         System.out.print("Ingrese una nota entre 0 y 10: ");
144         nota = input.nextInt();
145
146         if(nota < 0 || nota > 10){
147             System.out.println("Error: Nota invalida. Ingrese una nota entre 0 y 10");
148         }
149
150     } while(nota < 0 || nota > 10);
151
152     System.out.println("Nota guardada correctamente");
153 }
154
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >

Output - tp2 programacion estructurada (run) x

```
run:
Ingresa una nota entre 0 y 10: 15
Error: Nota invalida. Ingrese una nota entre 0 y 10
Ingresa una nota entre 0 y 10: -2
Error: Nota invalida. Ingrese una nota entre 0 y 10
Ingresa una nota entre 0 y 10: 8
Nota guardada correctamente
BUILD SUCCESSFUL (total time: 11 seconds)
```

Funciones:

8. Cálculo del Precio Final con impuesto y descuento. Crea un método `calcularPrecioFinal(double impuesto, double descuento)` que calcule el precio final de un producto en un e-commerce. La fórmula es:  $\text{PrecioFinal} = \text{PrecioBase} + (\text{PrecioBase} \times \text{Impuesto}) - (\text{PrecioBase} \times \text{Descuento})$

Desde `main()`, solicita el precio base del producto, el porcentaje de impuesto y el porcentaje de descuento, llama al método y muestra el precio final. Ejemplo de entrada/salida: Ingrese el precio base del producto: 100

Ingrese el impuesto en porcentaje (Ejemplo: 10 para 10%): 10

Ingrese el descuento en porcentaje (Ejemplo: 5 para 5%): 5

El precio final del producto es: 105.0

```
155 static void ejercicio_8(Scanner input){
156
157     System.out.print("Ingrese el precio base del producto: ");
158     double precioBase = Double.parseDouble(input.nextLine());
159     System.out.print("Ingrese el impuesto en porcentaje (Ejemplo: 10 para 10%): ");
160     double impuesto = Double.parseDouble(input.nextLine());
161     System.out.print("Ingrese el descuento en porcentaje (Ejemplo: 5 para 5%): ");
162     double descuento = Double.parseDouble(input.nextLine());
163
164     double precioFinal = calcularPrecioFinal(precioBase, impuesto, descuento);
165
166     double impuestoMonto = precioBase * (impuesto / 100.0);
167     double descuentoMonto = precioBase * (descuento / 100.0);
168
169     System.out.println("Precio base: " + precioBase);
170     System.out.println("Impuesto: " + impuesto + "% = " + impuestoMonto);
171     System.out.println("Descuento: " + descuento + "% = " + descuentoMonto);
172     System.out.println("Precio final: " + precioFinal);
173
174 }
175
176 static double calcularPrecioFinal(double precioBase, double impuestoPorcentaje, double descuentoPorcentaje){
177     double impuesto = impuestoPorcentaje / 100.0;
178     double descuento = descuentoPorcentaje / 100.0;
179     return precioBase + (precioBase * impuesto) - (precioBase * descuento);
180 }
181
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >

Output - tp2 programacion estructurada (run) x

```
run:
Ingrese el precio base del producto: 100
Ingrese el impuesto en porcentaje (Ejemplo: 10 para 10%): 10
Ingrese el descuento en porcentaje (Ejemplo: 5 para 5%): 5
Precio base: 100.0
Impuesto: 10.0% = 10.0
Descuento: 5.0% = 5.0
Precio final: 105.0
BUILD SUCCESSFUL (total time: 8 seconds)
```



9. Composición de funciones para calcular costo de envío y total de compra. a. `calcularCostoEnvio(double peso, String zona)`: Calcula el costo de envío basado en la zona de envío (Nacional o Internacional) y el peso del paquete. Nacional: \$5 por kg Internacional: \$10 por kg b. `calcularTotalCompra(double precioProducto, double costoEnvio)`: Usa `calcularCostoEnvio` para sumar el costo del producto con el costo de envío. Desde `main()`, solicita el peso del paquete, la zona de envío y el precio del producto. Luego, muestra el total a pagar. Ejemplo de entrada/salida: Ingrese el precio del producto: 50 Ingrese el peso del paquete en kg: 2 Ingrese la zona de envío (Nacional/Internacional): Nacional El costo de envío es: 10.0 El total a pagar es: 60.0

```
182 static void ejercicio_9(Scanner input){
183     double precioProducto, peso;
184     String zona;
185
186     System.out.print("Ingresa el precio del producto: ");
187     precioProducto = Double.parseDouble(input.nextLine());
188     System.out.print("Ingresa el peso del producto en kg: ");
189     peso = Double.parseDouble(input.nextLine());
190     System.out.print("Ingresa la zona de envio del producto: ");
191     zona = input.nextLine();
192
193     double costoEnvio = calcularCostoEnvio(peso, zona);
194     double total = calcularTotalCompra(precioProducto, costoEnvio);
195
196     System.out.println("El costo del envio es: " + costoEnvio);
197     System.out.println("El total a pagar es: " + total);
198
199 }
200
201 static double calcularCostoEnvio(double peso, String zona){
202
203     if(zona.equalsIgnoreCase("Nacional")){
204         return peso * 5.0;
205     }else if(zona.equalsIgnoreCase("Internacional")){
206         return peso * 10.0;
207     }else{
208         System.out.println("ERROR. Ingresa nacional o internacional");
209         return 0.0;
210     }
211 }
212
213 static double calcularTotalCompra(double precioProducto, double costoEnvio){
214
215     return precioProducto + costoEnvio;
216 }
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >

Output - tp2 programacion estructurada (run) X

```
run:
Ingresa el precio del producto: 50
Ingresa el peso del producto en kg: 2
Ingresa la zona de envio del producto: Nacional
El costo del envio es: 10.0
El total a pagar es: 60.0
BUILD SUCCESSFUL (total time: 11 seconds)
```

10. Actualización de stock a partir de venta y recepción de productos.

Crea un método actualizarStock(int stockActual, int cantidadVendida, int cantidadRecibida), que calcule el nuevo stock después de una venta y recepción de productos:

$$\text{NuevoStock} = \text{StockActual} - \text{CantidadVendida} + \text{CantidadRecibida}$$

$$\text{NuevoStock} = \text{CantidadVendida} + \text{CantidadRecibida}$$

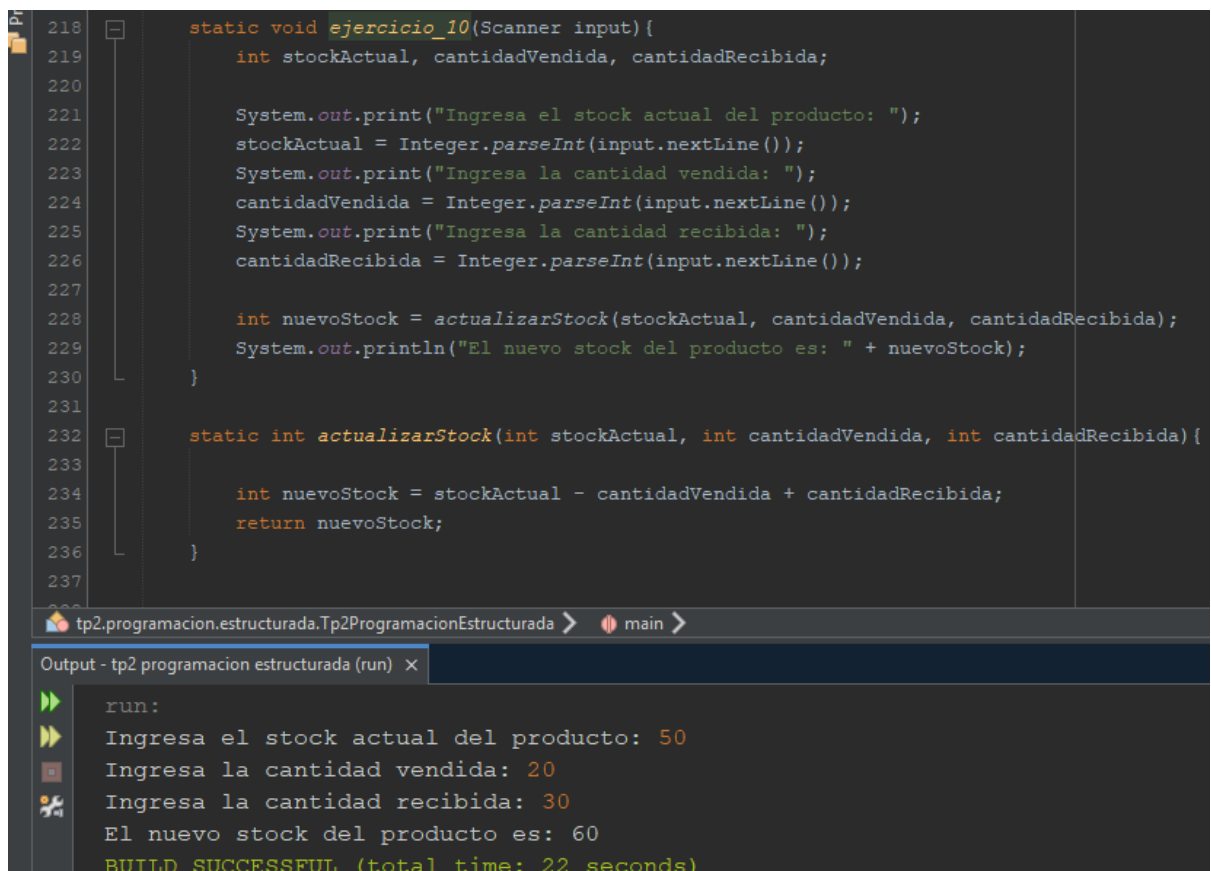
Desde main(), solicita al usuario el stock actual, la cantidad vendida y la cantidad recibida, y muestra el stock actualizado. Ejemplo de entrada/salida:

Ingrese el stock actual del producto: 50

Ingrese la cantidad vendida: 20

Ingrese la cantidad recibida: 30

El nuevo stock del producto es: 60



```
218 static void ejercicio_10(Scanner input) {
219     int stockActual, cantidadVendida, cantidadRecibida;
220
221     System.out.print("Ingresa el stock actual del producto: ");
222     stockActual = Integer.parseInt(input.nextLine());
223     System.out.print("Ingresa la cantidad vendida: ");
224     cantidadVendida = Integer.parseInt(input.nextLine());
225     System.out.print("Ingresa la cantidad recibida: ");
226     cantidadRecibida = Integer.parseInt(input.nextLine());
227
228     int nuevoStock = actualizarStock(stockActual, cantidadVendida, cantidadRecibida);
229     System.out.println("El nuevo stock del producto es: " + nuevoStock);
230 }
231
232 static int actualizarStock(int stockActual, int cantidadVendida, int cantidadRecibida) {
233
234     int nuevoStock = stockActual - cantidadVendida + cantidadRecibida;
235     return nuevoStock;
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
```

```
tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >
Output - tp2 programacion estructurada (run) x
run:
Ingresa el stock actual del producto: 50
Ingresa la cantidad vendida: 20
Ingresa la cantidad recibida: 30
El nuevo stock del producto es: 60
BUILD SUCCESSFUL (total time: 22 seconds)
```

## 11. Cálculo de descuento especial usando variable global.

Declara una variable global Ejemplo de entrada/salida: = 0.10. Luego, crea un método `calcularDescuentoEspecial(double precio)` que use la variable global para calcular el descuento especial del 10%.

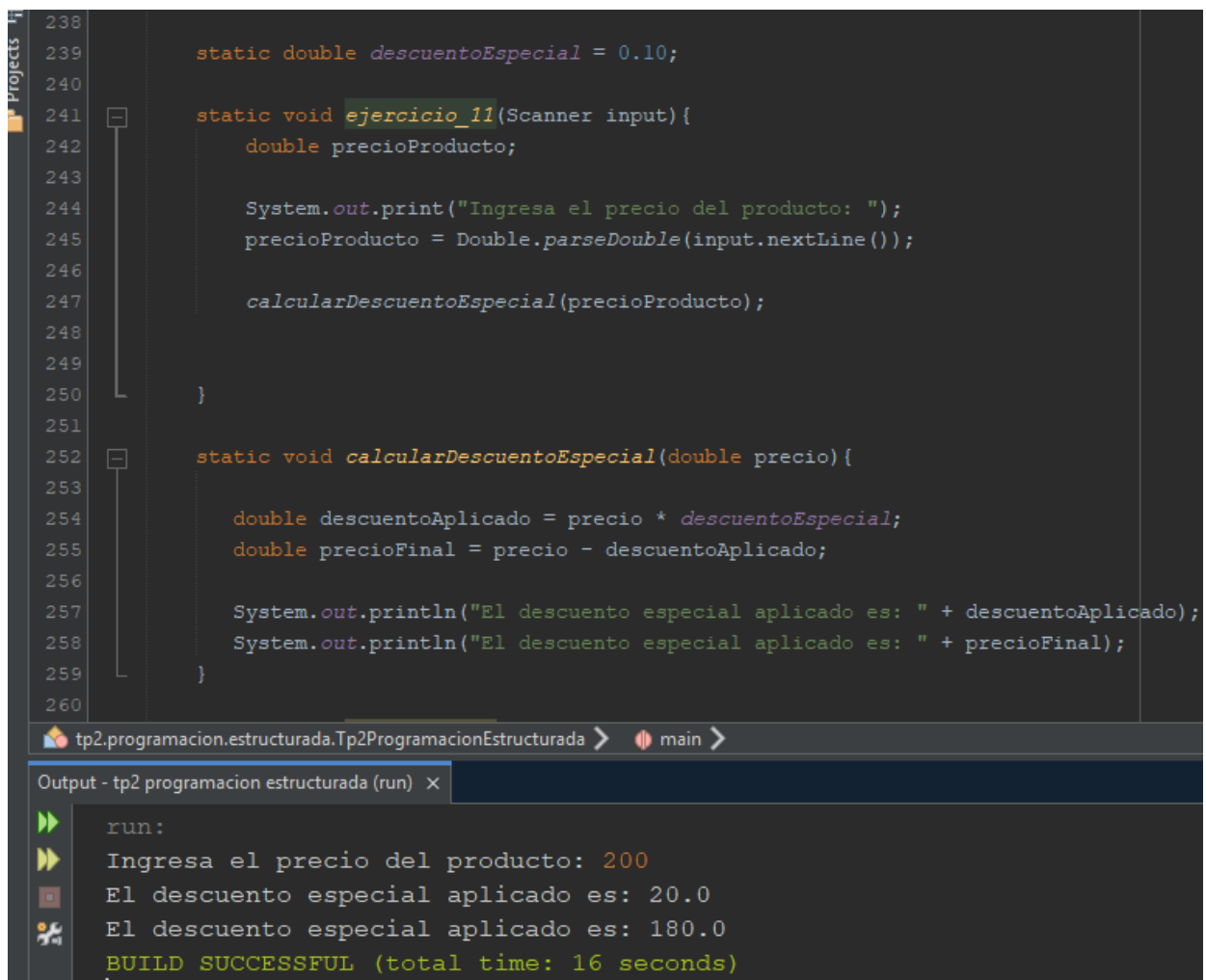
Dentro del método, declara una variable local `descuentoAplicado`, almacena el valor del descuento y muestra el precio final con descuento.

Ejemplo de entrada/salida:

Ingrese el precio del producto: 200

El descuento especial aplicado es: 20.0

El precio final con descuento es: 180.0



```
238
239     static double descuentoEspecial = 0.10;
240
241     static void ejercicio_11(Scanner input){
242         double precioProducto;
243
244         System.out.print("Ingresa el precio del producto: ");
245         precioProducto = Double.parseDouble(input.nextLine());
246
247         calcularDescuentoEspecial(precioProducto);
248
249     }
250
251
252     static void calcularDescuentoEspecial(double precio){
253
254         double descuentoAplicado = precio * descuentoEspecial;
255         double precioFinal = precio - descuentoAplicado;
256
257         System.out.println("El descuento especial aplicado es: " + descuentoAplicado);
258         System.out.println("El precio final con descuento es: " + precioFinal);
259     }
260
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > main >

Output - tp2 programacion estructurada (run) x

```
run:
Ingresa el precio del producto: 200
El descuento especial aplicado es: 20.0
El precio final con descuento es: 180.0
BUILD SUCCESSFUL (total time: 16 seconds)
```

Arrays y Recursividad: 12. Modificación de un array de precios y visualización de resultados. Crea un programa que: a. Declare e inicialice un array con los precios de algunos productos. b. Muestre los valores originales de los precios. c. Modifique el precio de un producto específico. d. Muestre los valores modificados. Salida esperada: Precios originales: Precio: \$199.99 Precio: \$299.5 Precio: \$149.75 Precio: \$399.0 Precio: \$89.99 Precios modificados: Precio: \$199.99 Precio: \$299.5 Precio: \$129.99 Precio: \$399.0 Precio: \$89.99

```
261 static void ejercicio_12(Scanner input){
262
263     double[] precios = {199.99, 2299.5, 149.75, 399.0, 89.99};
264
265     System.out.println("Precios originales: ");
266     for(int i = 0; i < precios.length; i++){
267         System.out.println("Precio: $" + precios[i]);
268     }
269
270     precios[2] = 129.99;
271
272     System.out.println("Precios modificados: ");
273     for(int i = 0; i < precios.length; i++){
274         System.out.println("Precio: $" + precios[i]);
275     }
276 }
277
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > ejercicio\_13 >

Output - tp2 programacion estructurada (run) x

```
run:
Precios originales:
Precio: $199.99
Precio: $2299.5
Precio: $149.75
Precio: $399.0
Precio: $89.99
Precios modificados:
Precio: $199.99
Precio: $2299.5
Precio: $129.99
Precio: $399.0
Precio: $89.99
BUILD SUCCESSFUL (total time: 0 seconds)
```

13. Impresión recursiva de arrays antes y después de modificar un elemento. Crea un programa que:

- Declare e inicialice un array con los precios de algunos productos.
- Use una función recursiva para mostrar los precios originales.
- Modifique el precio de un producto específico.
- Use otra función recursiva para mostrar los valores modificados.

Salida esperada: Precios originales: Precio: \$199.99 Precio: \$2299.5 Precio: \$149.75 Precio: \$399.0 Precio: \$89.99 Precios modificados: Precio: \$199.99 Precio: \$2299.5 Precio: \$129.99 Precio: \$399.0 Precio: \$89.99

```
278 static void ejercicio_13(Scanner input){
279
280     double[] precios = {199.99, 2299.5, 149.75, 399.0, 89.99};
281
282     System.out.println("Precios originales: ");
283     recursiva(precios, 0);
284
285     precios[2] = 129.99;
286
287     System.out.println("Precios modificados: ");
288     recursiva(precios, 0);
289
290 }
291
292 static void recursiva(double[] array, int indice){
293
294     if(indice < array.length){
295
296         System.out.println("Precio: $" + array[indice]);
297         recursiva(array, indice + 1);
298
299     }
```

tp2.programacion.estructurada.Tp2ProgramacionEstructurada > ejercicio\_13 >

Output - tp2 programacion estructurada (run) x

```
run:
Precios originales:
Precio: $199.99
Precio: $2299.5
Precio: $149.75
Precio: $399.0
Precio: $89.99
Precios modificados:
Precio: $199.99
Precio: $2299.5
Precio: $129.99
Precio: $399.0
Precio: $89.99
BUILD SUCCESSFUL (total time: 0 seconds)
```

Repo git: [https://github.com/FacuAuciello/UTN\\_JAVA\\_2do\\_Cuatri/tree/main](https://github.com/FacuAuciello/UTN_JAVA_2do_Cuatri/tree/main)