

Lesson 02 Demo 05

Sorting Arrays with Methods

Objective: Sort an array using methods and a bubble sort algorithm

Tools required: Eclipse IDE

Prerequisites: None

Steps to be followed:

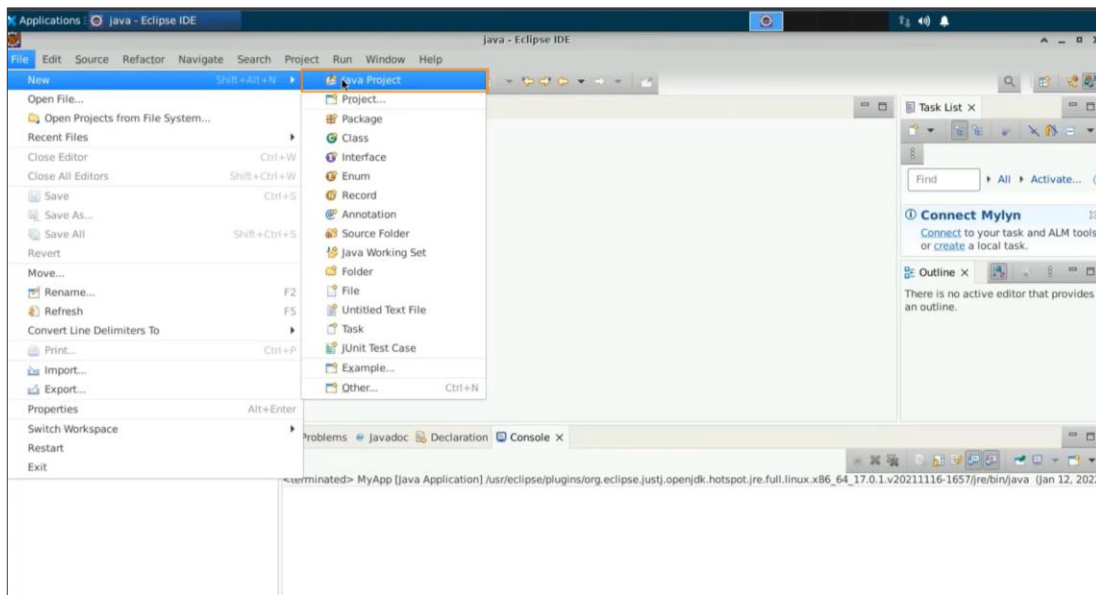
1. Create a class and a main method
2. Create a method of static void print to print the array
3. Execute the bubble sort method and reprint the array
4. Calculate the length of the array and swap values
5. Dry run the bubble sort algorithm and add a debugger view

Step 1: Create a class and a main method

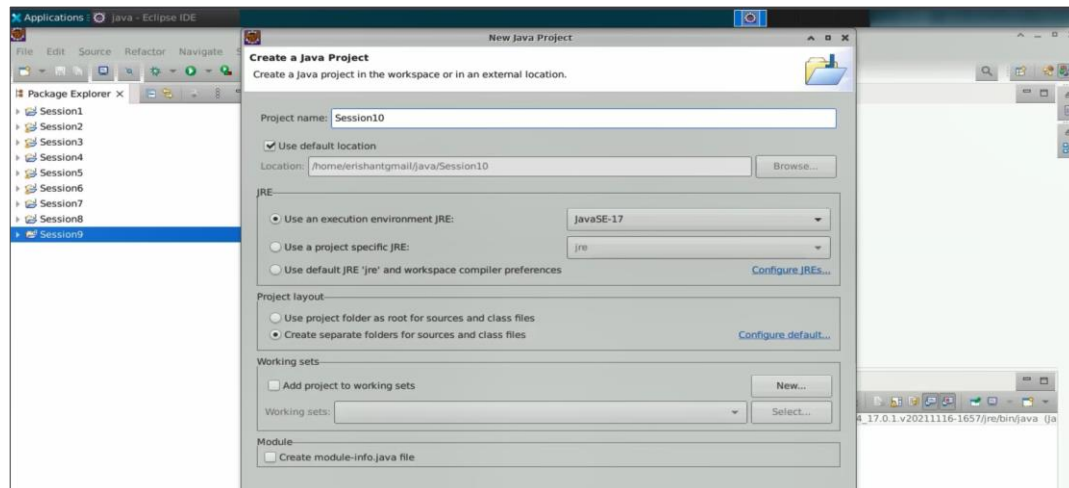
1.1 Open the Eclipse IDE



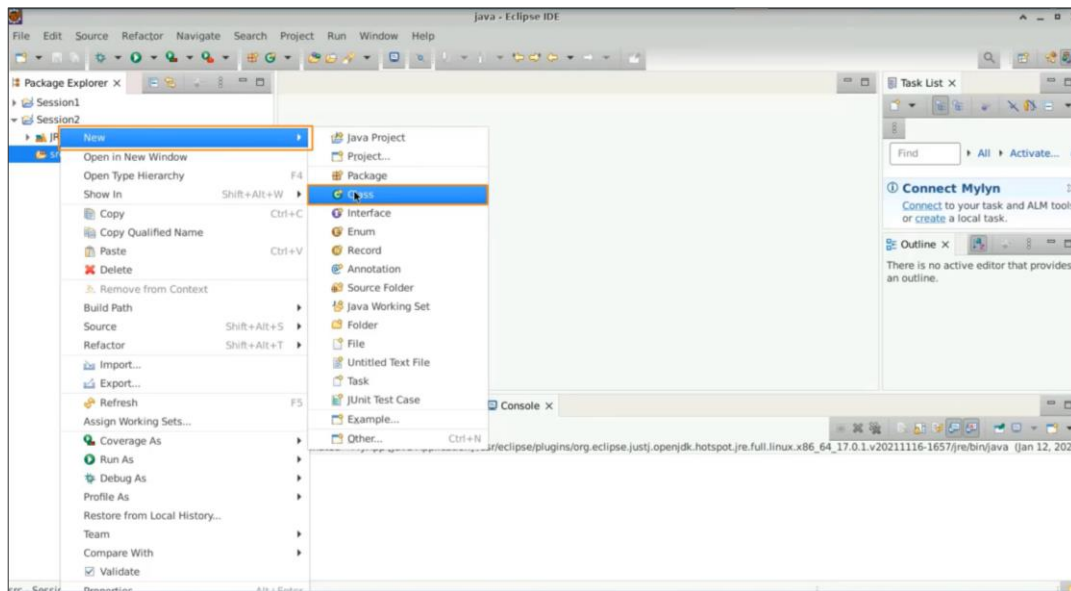
1.2. Select **File**, then **New**, and then **Java project**



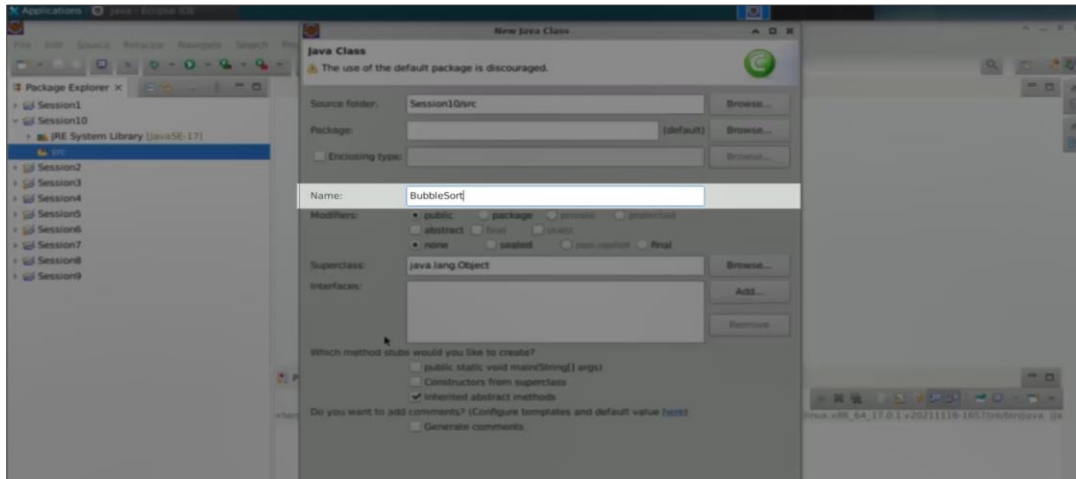
1.3 Name the project “**Session10**”, uncheck “**Create a module info dot Java file**”, and press **Finish**



1.4 With a **Session10** on the src, do a right-click and create a **new class**

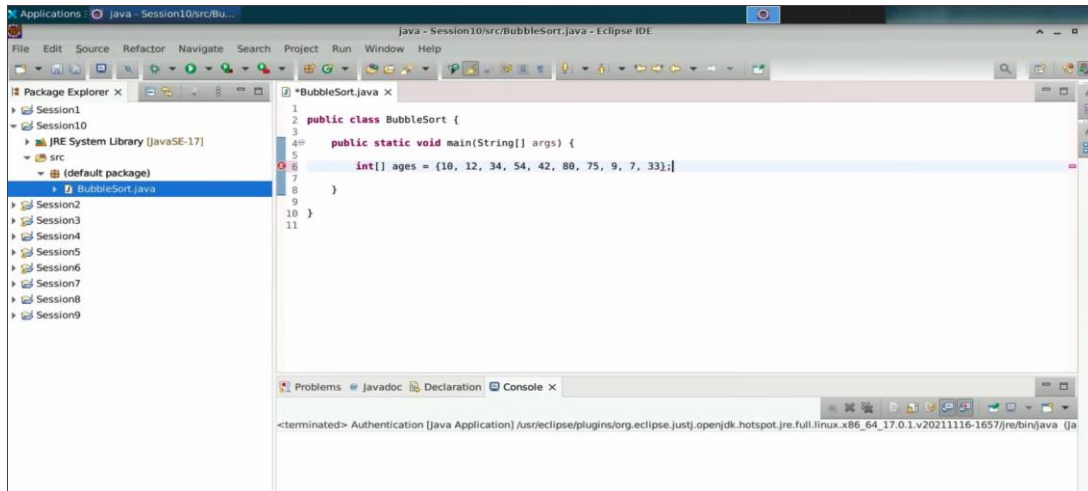


1.5 Name this class as an **BubbleSort**, then select the **main method**, and then select **finish**



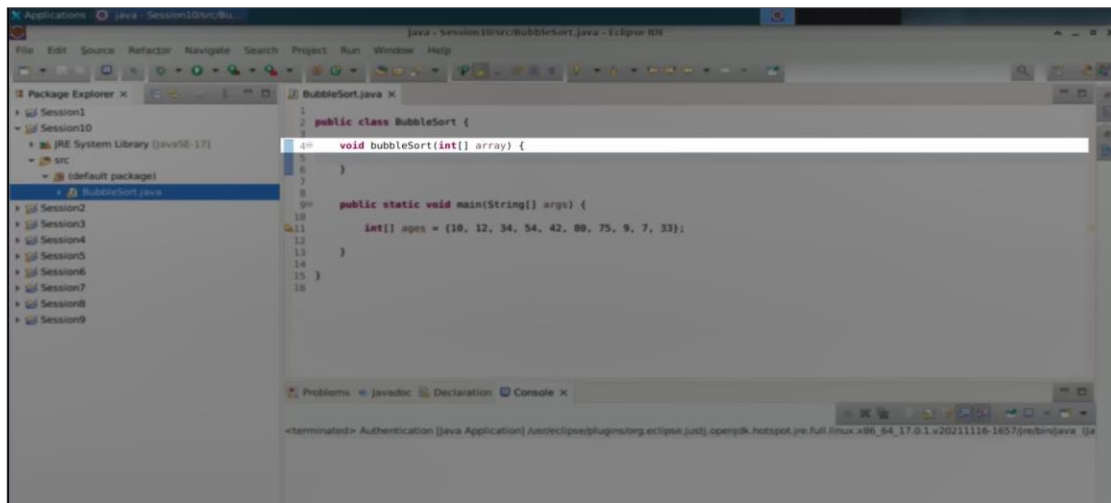
Step 2: Create a method of static void print to print the array

2.1 Consider you have an array named as ages. You have ten different entries



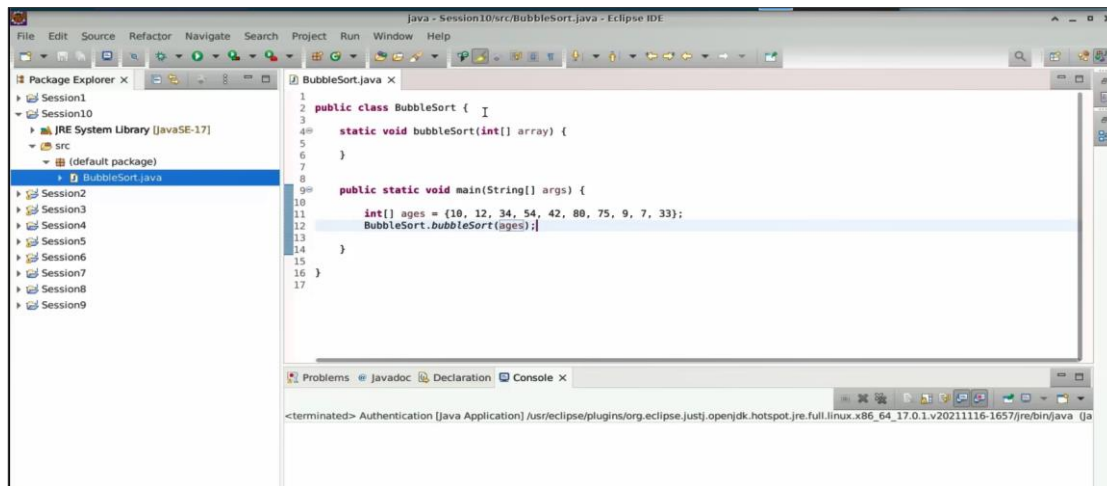
```
1 public class BubbleSort {
2
3
4     public static void main(String[] args) {
5         int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
6     }
7
8 }
9
10
11
```

2.2 Now you want this array to be sorted. The simplest one is the bubble sort. Write a method void bubble sort and inside it, you will take one array as input, which you wish to sort. Here you go with the bubble sort, a method with an array as input



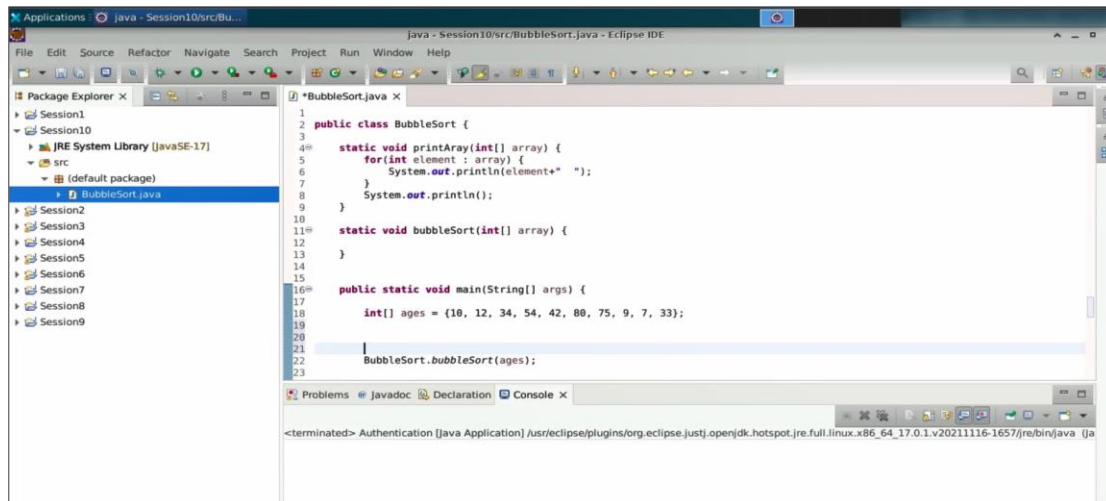
```
1 public class BubbleSort {
2
3
4     void bubbleSort(int[] array) {
5
6     }
7
8     public static void main(String[] args) {
9
10         int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
11     }
12
13 }
14
15
16
```

2.3 Make it static. Type bubble sort, and you'll execute this method called bubble sort. And you pass the edges as your array with the input to the method here

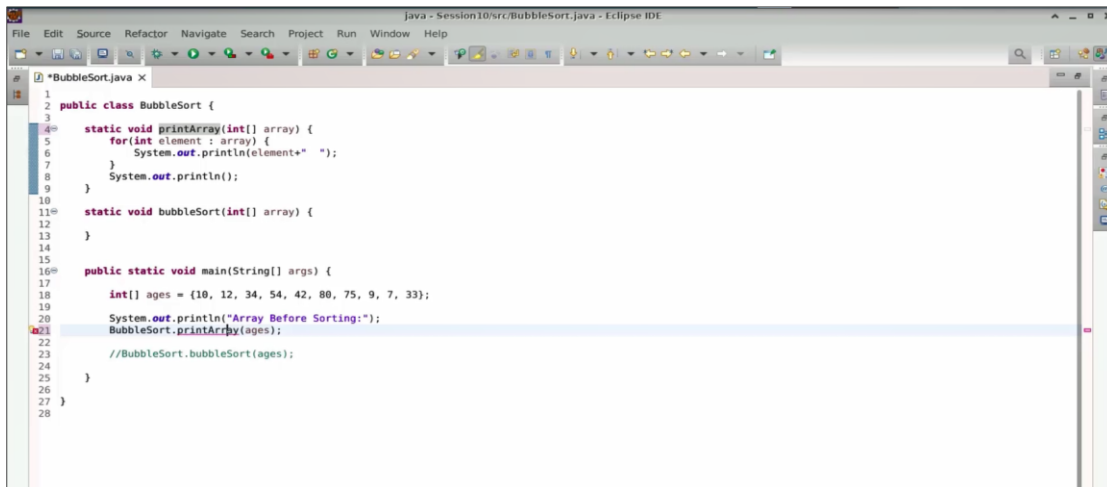


Step 3: Execute the bubble sort method and reprint the array

3.1 You will create another method called a static void print array. This method is also going to take one integer array as input. And type for integer element in the array. Print down this element with the space. Once the loop is finished, add an empty print line

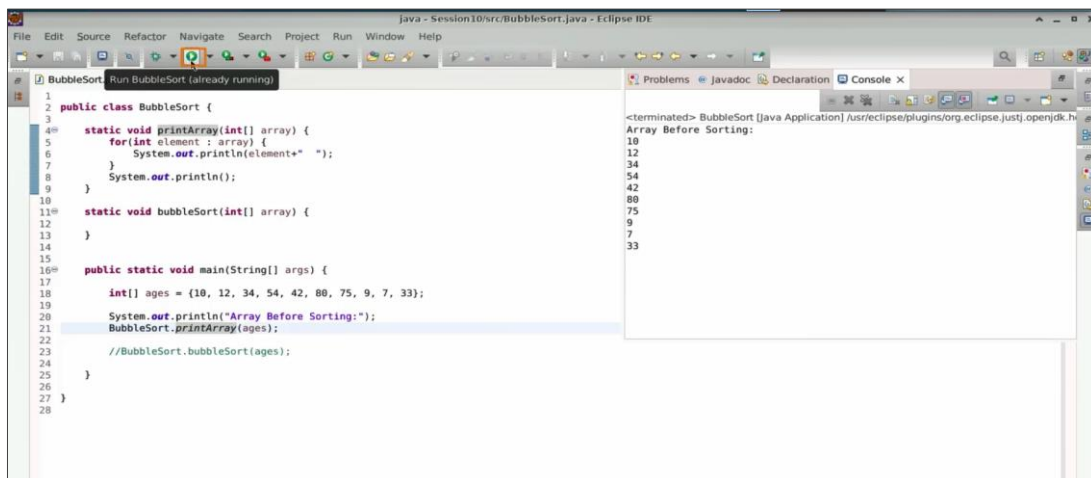


3.2 Type Array before sorting. You can add the class name that's the bubble sort dot print the array and you pass the ages, then just print the array



```
1 public class BubbleSort {
2
3
4     static void printArray(int[] array) {
5         for(int element : array) {
6             System.out.println(element+" ");
7         }
8         System.out.println();
9     }
10
11     static void bubbleSort(int[] array) {
12
13     }
14
15
16     public static void main(String[] args) {
17         int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
18
19         System.out.println("Array Before Sorting:");
20         BubbleSort.printArray(ages);
21
22         //BubbleSort.bubbleSort(ages);
23     }
24 }
25
26
27
28
```

3.3 Run the code and see what kind of order it is:

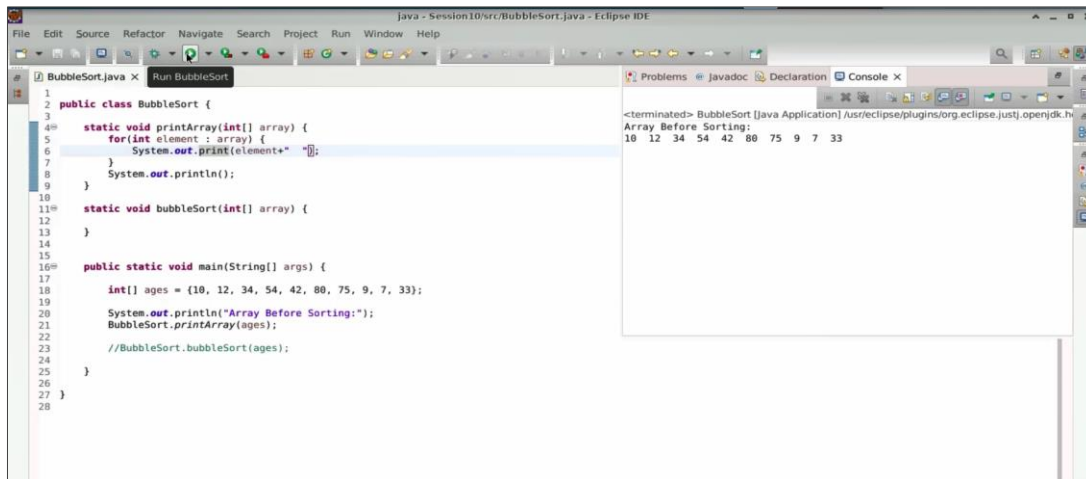


```
1 public class BubbleSort {
2
3
4     static void printArray(int[] array) {
5         for(int element : array) {
6             System.out.println(element+" ");
7         }
8         System.out.println();
9     }
10
11     static void bubbleSort(int[] array) {
12
13     }
14
15
16     public static void main(String[] args) {
17         int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
18
19         System.out.println("Array Before Sorting:");
20         BubbleSort.printArray(ages);
21
22         //BubbleSort.bubbleSort(ages);
23     }
24 }
25
26
27
28
```

Console Output:

```
<terminated> BubbleSort [Java Application] /usr/eclipse/plugins/org.eclipse.justi.openjdk.hotspot.jre.full/bin/linux64/java
Array Before Sorting:
10
12
34
54
42
80
75
9
7
33
```

3.4 Type print, not the print line, let it be printed in the same line, so you can see 10 12 34 54.
It is exactly in the same order in which you have created an array



```

1 public class BubbleSort {
2
3     static void printArray(int[] array) {
4         for(int element : array) {
5             System.out.print(element+" ");
6         }
7         System.out.println();
8     }
9
10    static void bubbleSort(int[] array) {
11    }
12
13    public static void main(String[] args) {
14
15        int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
16
17        System.out.println("Array Before Sorting:");
18        BubbleSort.printArray(ages);
19        //BubbleSort.bubbleSort(ages);
20    }
21 }
22
23
24
25
26
27
28

```

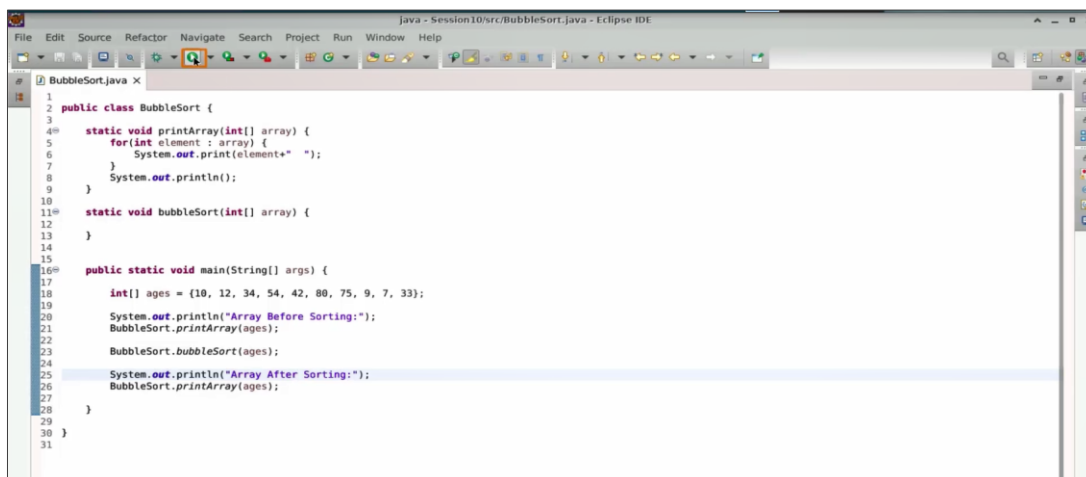
Console Output:

```

<terminated> BubbleSort [Java Application] /usr/eclipse/plugins/org.eclipse.jst.openjdk.h
Array Before Sorting:
10 12 34 54 42 80 75 9 7 33

```

3.5 Now, to execute this method called **bubbleSort.bubbleSort()**, once you execute this method, you can print the array after sorting to see the changes

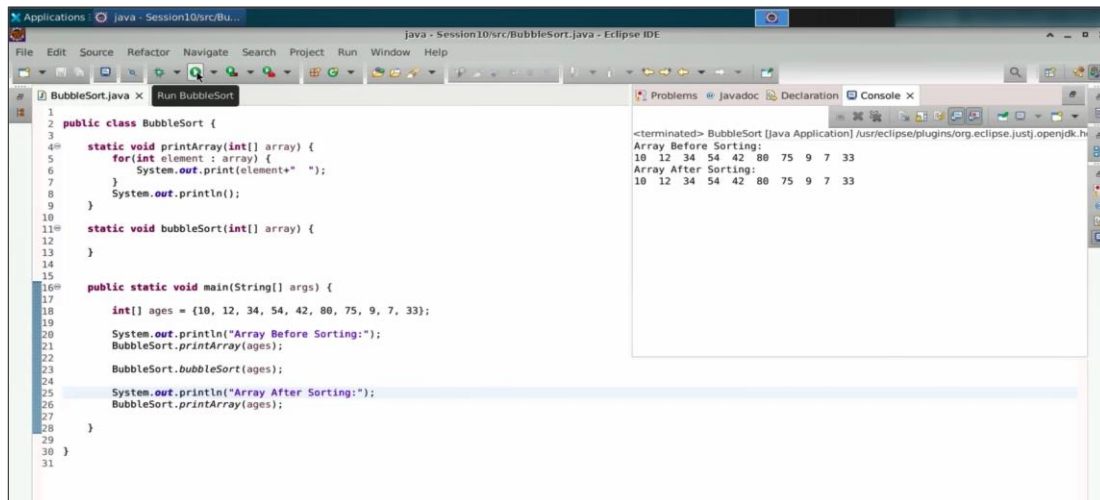


```

1 public class BubbleSort {
2
3     static void printArray(int[] array) {
4         for(int element : array) {
5             System.out.print(element+" ");
6         }
7         System.out.println();
8     }
9
10    static void bubbleSort(int[] array) {
11    }
12
13    public static void main(String[] args) {
14
15        int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
16
17        System.out.println("Array Before Sorting:");
18        BubbleSort.printArray(ages);
19        BubbleSort.bubbleSort(ages);
20        System.out.println("Array After Sorting:");
21        BubbleSort.printArray(ages);
22    }
23 }
24
25
26
27
28
29
30
31

```

3.6 Run the code and you see is that before sorting and after sorting, the output is exactly same. That is for reason why you have not implemented the bubble sort method yet



The screenshot shows the Eclipse IDE with a Java project. The main editor displays the `BubbleSort.java` file. The code defines a `BubbleSort` class with two static methods: `printArray` and `bubbleSort`. The `main` method creates an array of ages and prints it before and after calling `bubbleSort`. The console output shows the array before and after sorting, which is identical, indicating that the `bubbleSort` method has not yet been implemented.

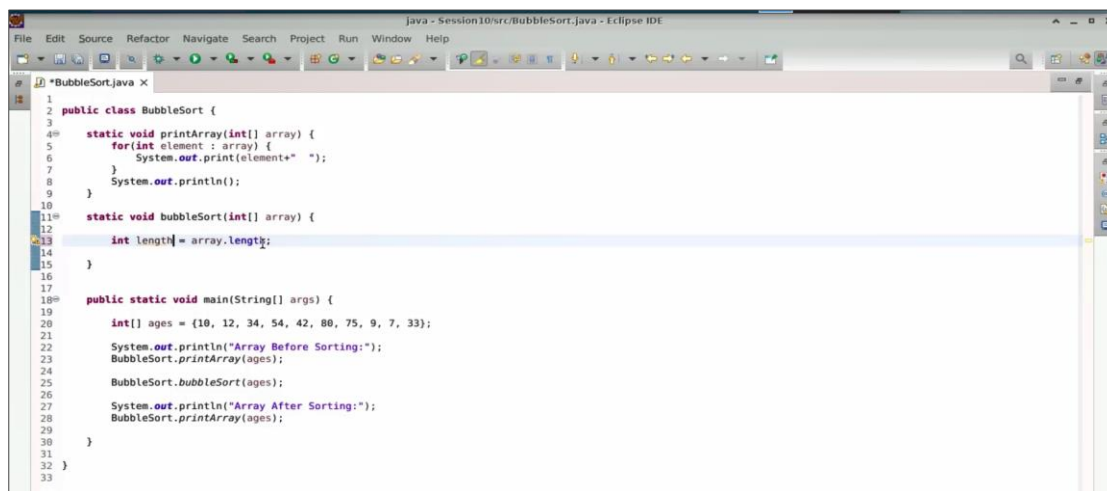
```
1 public class BubbleSort {
2
3
4     static void printArray(int[] array) {
5         for(int element : array) {
6             System.out.print(element+" ");
7         }
8         System.out.println();
9     }
10
11     static void bubbleSort(int[] array) {
12     }
13
14
15
16     public static void main(String[] args) {
17
18         int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
19
20         System.out.println("Array Before Sorting:");
21         BubbleSort.printArray(ages);
22
23         BubbleSort.bubbleSort(ages);
24
25         System.out.println("Array After Sorting:");
26         BubbleSort.printArray(ages);
27     }
28 }
29
30
31
```

Console Output:

```
<terminated> BubbleSort [Java Application] /usr/eclipse/plugins/org.eclipse.justi.openjdk.hotspot.jre.full.j2sdk-11.0.2-linux-x86_64/bin/java
Array Before Sorting:
10 12 34 54 42 80 75 9 7 33
Array After Sorting:
10 12 34 54 42 80 75 9 7 33
```

Step 4: Calculate the length of the array and swap values

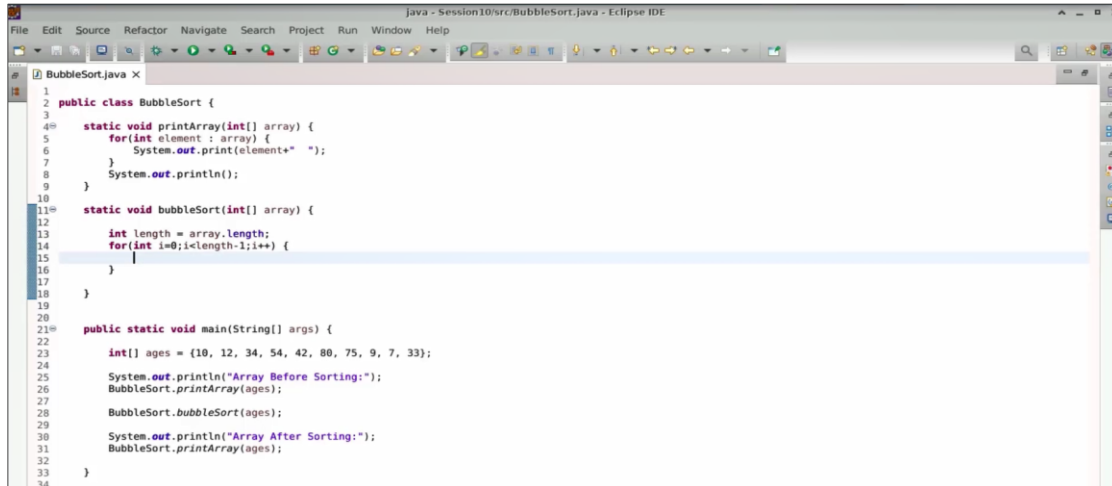
4.1 Now let us get started with the bubble sort method. The first thing you are going to do is, capture the length of the array in a length variable that you have created now



The screenshot shows the Eclipse IDE with the `BubbleSort.java` file. The `bubbleSort` method has been updated to include a line that captures the length of the array into a `length` variable.

```
1 public class BubbleSort {
2
3
4     static void printArray(int[] array) {
5         for(int element : array) {
6             System.out.print(element+" ");
7         }
8         System.out.println();
9     }
10
11     static void bubbleSort(int[] array) {
12         int length = array.length;
13     }
14
15
16
17     public static void main(String[] args) {
18
19         int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
20
21         System.out.println("Array Before Sorting:");
22         BubbleSort.printArray(ages);
23
24         BubbleSort.bubbleSort(ages);
25
26         System.out.println("Array After Sorting:");
27         BubbleSort.printArray(ages);
28     }
29 }
30
31
32
33
```


4.2 Now, you are going to write a for loop that starts with the i value as zero. The loop runs while i is less than length - 1, and increments i by one. This loop begins from the first index and continues until it is less than length - 1.

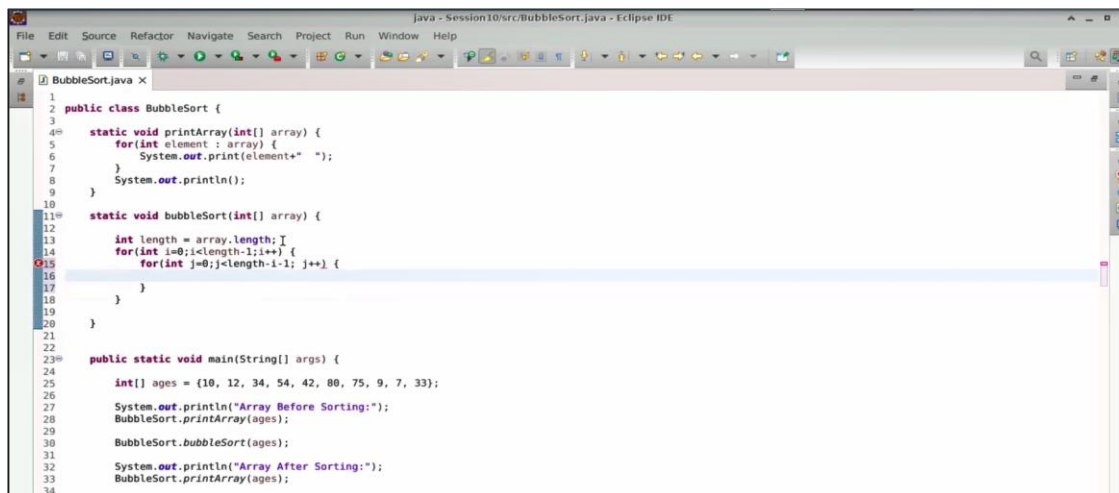


```

1 public class BubbleSort {
2
3     static void printArray(int[] array) {
4         for(int element : array) {
5             System.out.print(element+" ");
6         }
7         System.out.println();
8     }
9
10    static void bubbleSort(int[] array) {
11
12        int length = array.length;
13        for(int i=0; i<length-1; i++) {
14
15        }
16    }
17
18
19
20    public static void main(String[] args) {
21
22        int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
23
24        System.out.println("Array Before Sorting:");
25        BubbleSort.printArray(ages);
26
27        BubbleSort.bubbleSort(ages);
28
29        System.out.println("Array After Sorting:");
30        BubbleSort.printArray(ages);
31
32    }
33
34 }

```

4.3 Inside this you are going to create another for loop which goes as j value 0. And the j will be limited to the length-i-1 and then give j++. Now length-i-1 would typically decrease. The i variable from the length and then one more -1. If you begin with the zero, so the loop is going to go from zero till 8 and skip 9

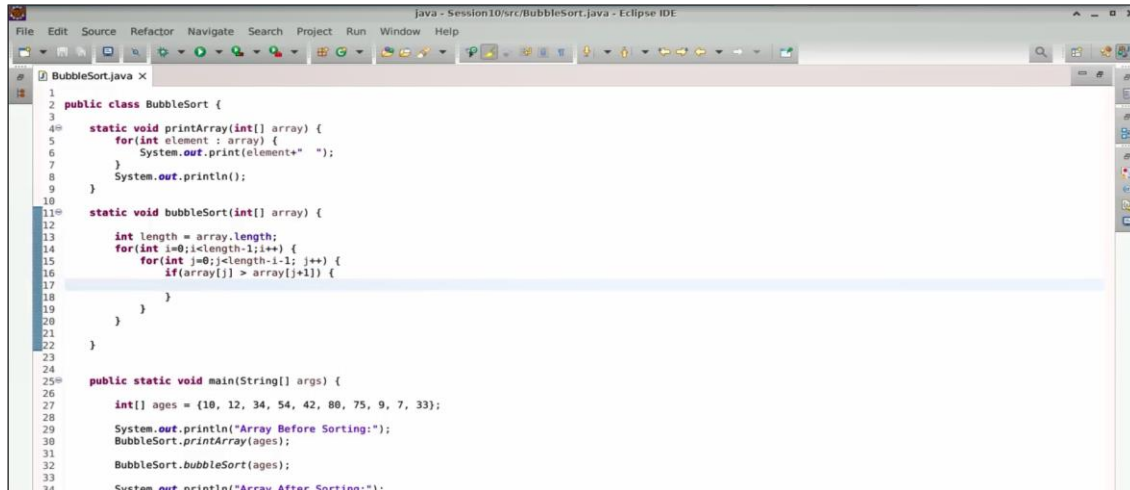


```

1 public class BubbleSort {
2
3     static void printArray(int[] array) {
4         for(int element : array) {
5             System.out.print(element+" ");
6         }
7         System.out.println();
8     }
9
10    static void bubbleSort(int[] array) {
11
12        int length = array.length;
13        for(int i=0; i<length-1; i++) {
14            for(int j=0; j<length-i-1; j++) {
15
16            }
17        }
18    }
19
20
21
22    public static void main(String[] args) {
23
24        int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
25
26        System.out.println("Array Before Sorting:");
27        BubbleSort.printArray(ages);
28
29        BubbleSort.bubbleSort(ages);
30
31        System.out.println("Array After Sorting:");
32        BubbleSort.printArray(ages);
33
34    }
35 }

```

4.4 Now, inside the loop, add an if-statement: if **array[j]** is greater than **array[j+1]**. This is a standard algorithm. If an element is greater than the next element, you need to swap them. To perform the swap, you need to create a temporary variable

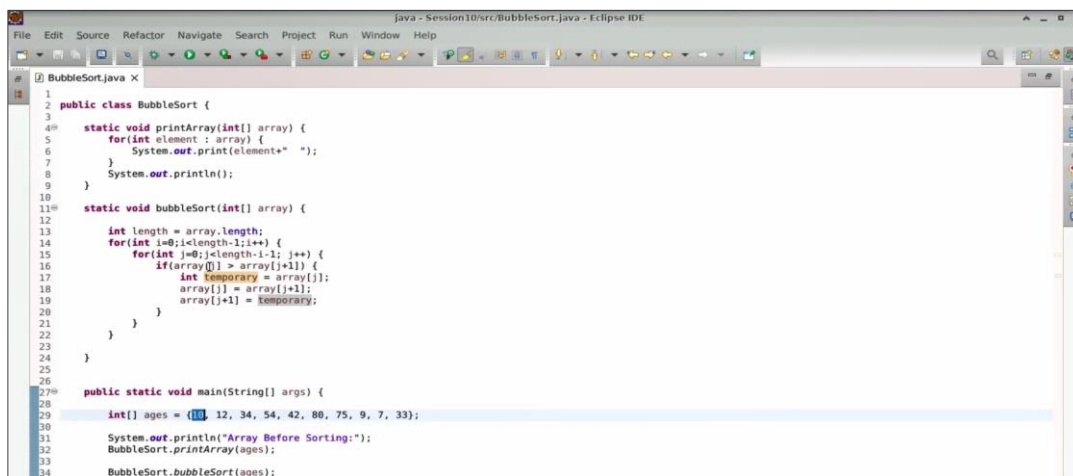


```

1 public class BubbleSort {
2
3
4     static void printArray(int[] array) {
5         for(int element : array) {
6             System.out.print(element+" ");
7         }
8         System.out.println();
9     }
10
11     static void bubbleSort(int[] array) {
12
13         int length = array.length;
14         for(int i=0; i<length-1; i++) {
15             for(int j=0; j<length-i-1; j++) {
16                 if(array[j] > array[j+1]) {
17
18                 }
19             }
20         }
21     }
22 }
23
24 public static void main(String[] args) {
25
26     int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
27
28     System.out.println("Array Before Sorting:");
29     BubbleSort.printArray(ages);
30
31     BubbleSort.bubbleSort(ages);
32
33     System.out.println("Array After Sorting:");
34 }

```

4.5 Let us add a temporary variable in which you will copy the array of j and then add the array of j that is j+1. And then in the array of j+1, you can copy your temporary. Now you can perform the swapping in much more different ways. But to use the swap concept with a very basic approach, the temporary variable which will hold the value of array of j



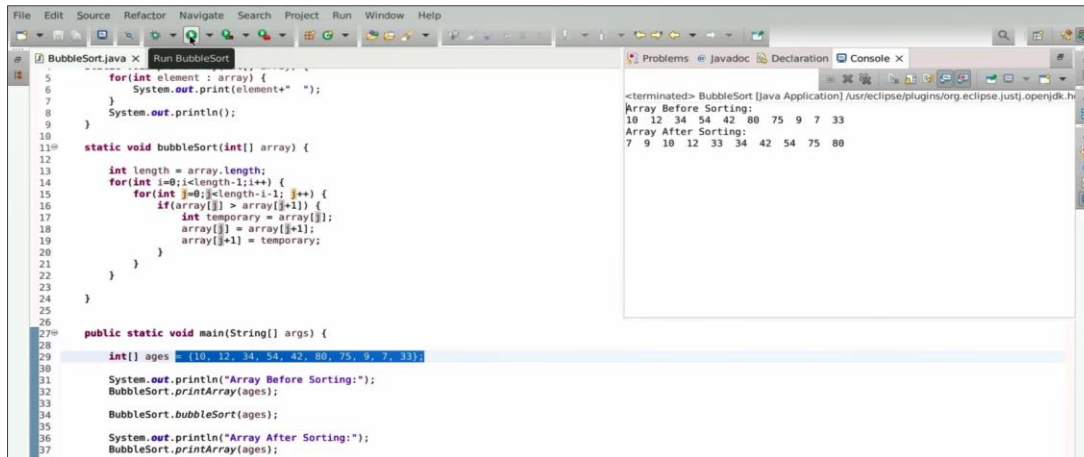
```

1 public class BubbleSort {
2
3
4     static void printArray(int[] array) {
5         for(int element : array) {
6             System.out.print(element+" ");
7         }
8         System.out.println();
9     }
10
11     static void bubbleSort(int[] array) {
12
13         int length = array.length;
14         for(int i=0; i<length-1; i++) {
15             for(int j=0; j<length-i-1; j++) {
16                 if(array[j] > array[j+1]) {
17                     int temporary = array[j];
18                     array[j] = array[j+1];
19                     array[j+1] = temporary;
20                 }
21             }
22         }
23     }
24 }
25
26 public static void main(String[] args) {
27
28     int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
29
30     System.out.println("Array Before Sorting:");
31     BubbleSort.printArray(ages);
32
33     BubbleSort.bubbleSort(ages);
34 }

```

Step 5: Dry run the bubble sort algorithm and add a debugger view

5.1 Let us run this algorithm. Now here you are with 7, 9, 10, 12, 13, 34 42, 54, 75 and 80. This algorithm is implemented well

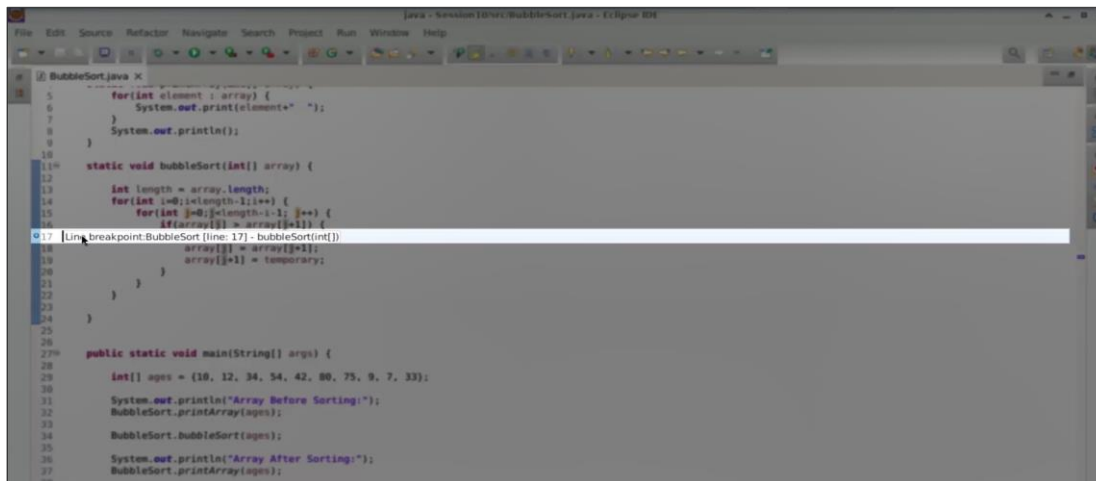


```
File Edit Source Refactor Navigate Search Project Run Window Help
BubbleSort.java x Run BubbleSort
5 for(int element : array) {
6     System.out.print(element+" ");
7 }
8 System.out.println();
9 }
10
11 static void bubbleSort(int[] array) {
12     int length = array.length;
13     for(int i=0; i<length-1; i++) {
14         for(int j=0; j<length-i-1; j++) {
15             if(array[j] > array[j+1]) {
16                 int temporary = array[j];
17                 array[j] = array[j+1];
18                 array[j+1] = temporary;
19             }
20         }
21     }
22 }
23
24 public static void main(String[] args) {
25     int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
26
27     System.out.println("Array Before Sorting:");
28     BubbleSort.printArray(ages);
29
30     BubbleSort.bubbleSort(ages);
31
32     System.out.println("Array After Sorting:");
33     BubbleSort.printArray(ages);
34 }
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
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
```

Console Output:

```
<terminated> BubbleSort [Java Application] /usr/eclipse/plugins/org.eclipse.justi.openjdk.hotspot.jre.full/bin/linux64/java
Array Before Sorting:
10 12 34 54 42 80 75 9 7 33
Array After Sorting:
7 9 10 12 33 34 42 54 75 80
```

5.2 Put a breakpoint on the if condition and the temporary variable



```
File Edit Source Refactor Navigate Search Project Run Window Help
BubbleSort.java x
5 for(int element : array) {
6     System.out.print(element+" ");
7 }
8 System.out.println();
9 }
10
11 static void bubbleSort(int[] array) {
12     int length = array.length;
13     for(int i=0; i<length-1; i++) {
14         for(int j=0; j<length-i-1; j++) {
15             if(array[j] > array[j+1]) {
16                 int temporary = array[j];
17                 array[j] = array[j+1];
18                 array[j+1] = temporary;
19             }
20         }
21     }
22 }
23
24 public static void main(String[] args) {
25     int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
26
27     System.out.println("Array Before Sorting:");
28     BubbleSort.printArray(ages);
29
30     BubbleSort.bubbleSort(ages);
31
32     System.out.println("Array After Sorting:");
33     BubbleSort.printArray(ages);
34 }
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
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
```

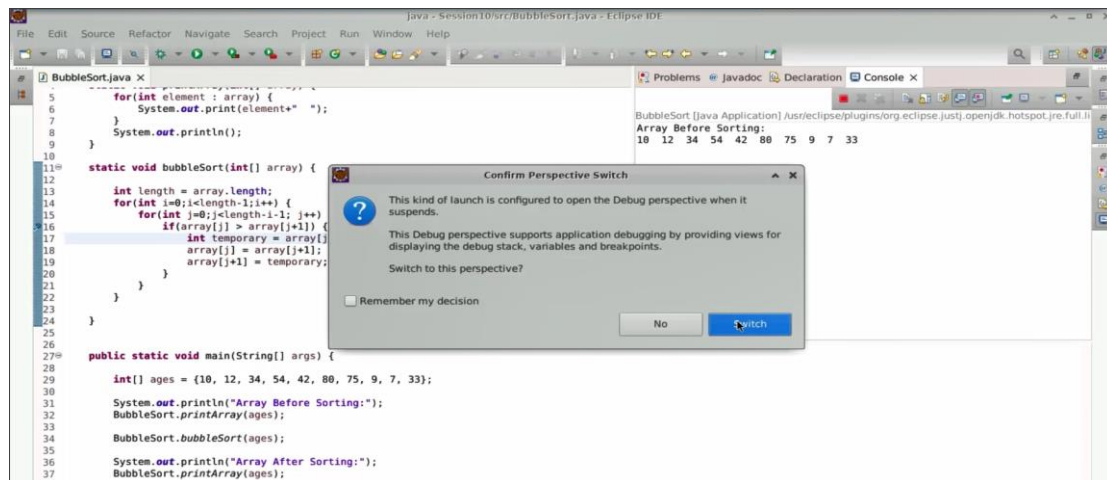
5.3 And now instead of selecting run, you can select debug the bubble sort

```

1  java - Session10/src/BubbleSort.java - Eclipse IDE
2  File Edit Source Refactor Navigate Search Project Run Window Help
3
4  BubbleSort.java
5  for(int element : array) {
6      System.out.print(element+ " ");
7  }
8  System.out.println();
9  }
10
11  static void bubbleSort(int[] array) {
12      int length = array.length;
13      for(int i=0; i<length-1; i++) {
14          for(int j=0; j<length-i-1; j++) {
15              if(array[j] > array[j+1]) {
16                  int temporary = array[j];
17                  array[j] = array[j+1];
18                  array[j+1] = temporary;
19              }
20          }
21      }
22  }
23
24  public static void main(String[] args) {
25      int[] ages = {10, 12, 34, 54, 42, 80, 75, 9, 7, 33};
26
27      System.out.println("Array Before Sorting:");
28      BubbleSort.printArray(ages);
29
30      BubbleSort.bubbleSort(ages);
31
32      System.out.println("Array After Sorting:");
33      BubbleSort.printArray(ages);
34  }
35
36
37
38

```

5.4 You are going to open other debugging perspectives



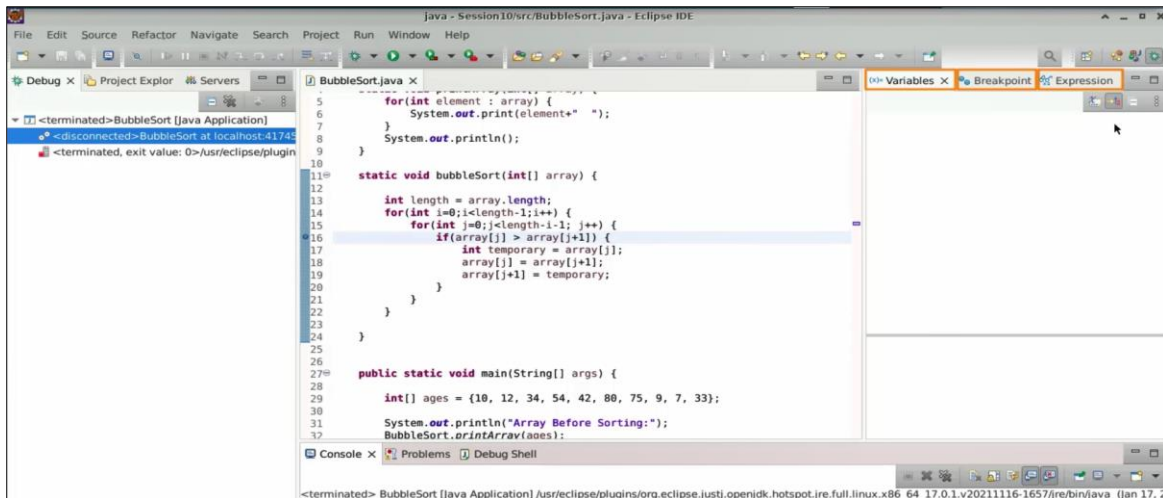
5.5 When you are executing this program, the array is there right with the data having the zero till the 9th index. The length is 10, i is zero and j is 0 at this point. These are the variables and you do have the values

Name	Value
array[1]	12
array[2]	34
array[3]	54
array[4]	42
array[5]	80
array[6]	75
array[7]	9
array[8]	7
array[9]	33
length	10
i	0
j	0

5.6 Let us resume now. The j value is 1, and then it becomes 2, leading to swaps and manipulations within the array. Wherever a swap occurs, the updated value of the array is displayed. In the very first pass, the largest element automatically moves to the end

Name	Value
array[1]	12
array[2]	34
array[3]	42
array[4]	54
array[5]	75
array[6]	80
array[7]	9
array[8]	7
array[9]	33
length	10
i	0
j	6

5.7 All these algorithmic approaches, you must observe through the debugger attached. You can simply put a line break and rather than executing your program normally, you can simply attach a debugger view. Remember, it is a debug explorer, and you will have variable breakpoints or the expressions shown on the right-hand side. You can see here; that the perspective is the debug perspective



By following the above steps, you have successfully sorted an array using methods and a bubble sort algorithm.