

Obligatorisk 2 – Sortering mm

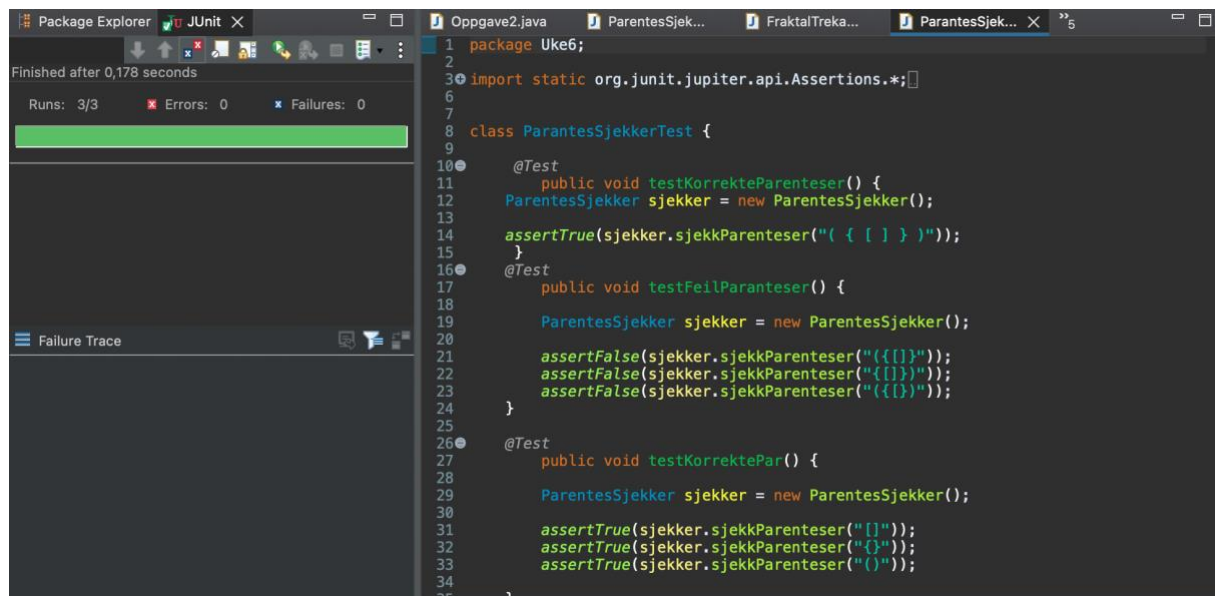
Marius Larsen Arnevik

Julie Elea Fjellstad

Ingvild Sirnes

Skjermbilder av kjøring av programmene

Oppgave uke 6 (Parentessjekker)



The screenshot shows an IDE with the following components:

- Package Explorer:** Shows the project structure with 'Uke6' and 'ParantesSjekkerTest'.
- JUnit Console:** Displays the test results: 'Finished after 0,178 seconds', 'Runs: 3/3', 'Errors: 0', and 'Failures: 0'. A green progress bar indicates successful completion.
- Failure Trace:** Empty, as all tests passed.
- Code Editor:** Contains the source code for 'Uke6.java' and 'ParantesSjekkerTest.java'.

```
1 package Uke6;
2
3 import static org.junit.jupiter.api.Assertions.*;
4
5
6
7
8 class ParantesSjekkerTest {
9
10     @Test
11     public void testKorrekteParenteser() {
12         ParentesSjekker sjekker = new ParentesSjekker();
13
14         assertTrue(sjekker.sjekkParenteser("( { [ ] } )"));
15     }
16
17     @Test
18     public void testFeilParanteser() {
19         ParentesSjekker sjekker = new ParentesSjekker();
20
21         assertFalse(sjekker.sjekkParenteser("{[[]}" ));
22         assertFalse(sjekker.sjekkParenteser("{[]}" ));
23         assertFalse(sjekker.sjekkParenteser("{[[]}" ));
24     }
25
26     @Test
27     public void testKorrektePar() {
28
29         ParentesSjekker sjekker = new ParentesSjekker();
30
31         assertTrue(sjekker.sjekkParenteser("[ ]"));
32         assertTrue(sjekker.sjekkParenteser("{}"));
33         assertTrue(sjekker.sjekkParenteser("{}"));
34     }
35 }
```

Oppgave uke 7 (Sorteringsvarianter)

```
Oppgave1.java  *Oppgave2.java X  Queue_Øving...  selectionSo...  »1
142  Integer[] tabell = generateTabell(nverdi,
143
144      System.out.println("\nN = " + nverdi);
145
146      maalSorteringsTid(tabell, "Insertion Sort", "O(n^2)");
147      maalSorteringsTid(tabell, "Selection Sort", "O(n^2)");
148      maalSorteringsTid(tabell, "Merge Sort", "O(n log n)");
149      maalSorteringsTid(tabell, "Quick Sort", "O(n log n)");
150  }
151  }
152  }
153
154  // quicksort og mergesort er O(n log n)
155  // insertion og selection er O(n^2)
156  // som en ser på utskriften er insertion og selection langt tregere enn
157  // quicksort og merge når n blir større
```

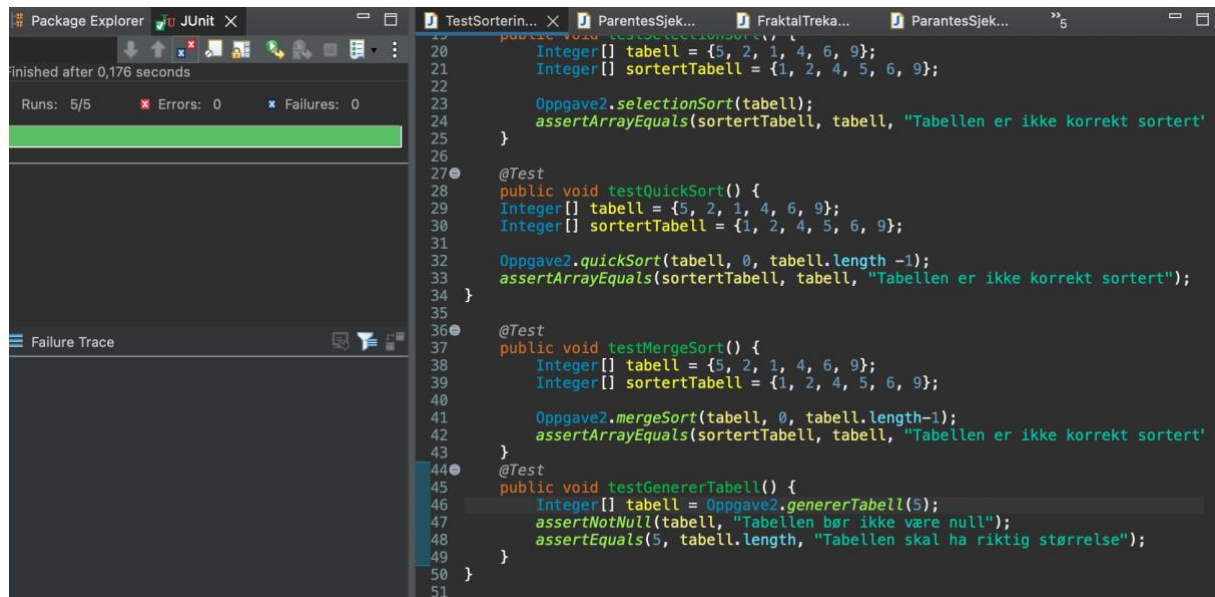
Console X

<terminated> Oppgave2 [Java Application] /Applications/Eclipse.app/Contents/Eclipse/plugins/org.eclipse.justj.openjdk.hot

```
N = 32000
Insertion Sort | Notasjon: O(n^2) | Tok: 1794.0 ms
Selection Sort | Notasjon: O(n^2) | Tok: 657.0 ms
Merge Sort | Notasjon: O(n log n) | Tok: 39.0 ms
Quick Sort | Notasjon: O(n log n) | Tok: 30.0 ms

N = 64000
Insertion Sort | Notasjon: O(n^2) | Tok: 3995.0 ms
Selection Sort | Notasjon: O(n^2) | Tok: 2588.0 ms
Merge Sort | Notasjon: O(n log n) | Tok: 17.0 ms
Quick Sort | Notasjon: O(n log n) | Tok: 18.0 ms

N = 128000
Insertion Sort | Notasjon: O(n^2) | Tok: 21929.0 ms
Selection Sort | Notasjon: O(n^2) | Tok: 14263.0 ms
Merge Sort | Notasjon: O(n log n) | Tok: 36.0 ms
Quick Sort | Notasjon: O(n log n) | Tok: 43.0 ms
```



Oppgave b

Får stackoverflow error når quicksort kjøres med 32000 like tall. Kjøretiden blir større fordi quicksort sorterer venstre og høyre side ved å se hvilke elementer som til venstre er mindre eller lik pivoten, og til høyre elementene som er større enn pivoten. Når alle elementene er like tar dette lang tid. Kjøretiden går fra $O(n \log n)$ til $O(n^2)$, som er worst case scenario for quicksort.