Løsningsforslag
Eksamen i ITF20006 Algoritmer og datastrukturer, 12.05.2022

Oppgave 1

A: O(n)
B: O(n)
C: $O(n^2)$
D: O(log n)
E: O(n·log n)

Oppgave 2

A: Rekkefølgen av tallene…vil være snudd om
B: Stacken vil være uforandret
C: Fibonaccitallene skrives ut stigende
D: 6
E: (a((bc)d)

Oppgave 3

A: Quicksort
B: Heapsort
C: Innstikksortering
D: Utplukksortering
E: Shell sort

Oppgave 4

A: O(n) for alle tre operasjoner
B: O(log n) for alle tre operasjoner
C: 7 5 8 1 6 9 0 3 2 4
D: 7 4 8 1 6 9 0 3 2
E: 15 10 23 25 20 35 42 39 30

Oppgave 5

A: (a)
B: 8 10 14 30 16 20 22
C: 14 13 12 8 10
D: 4
E: O(n)

Oppgave 6

A: (c)
B: Last-Come-First-Served Hashing
C: Hashing med kjeding
D: L = 2.0
E: Kvadratisk probing

Oppgave 7

A: Både BFS og DFS
B: Alle elementer i løsningsmatrisen vil være TRUE
C: DFS 0 1 2 3 4, BFS 0 1 3 4 2
D: 0 1 2 3 4
E: -1 0 1 2 0

Oppgave 8

a)
```
void settInn(String ord, int antall)
{
    Node ny = new Node(ord, antall);
    if (rot == null)
    {
       rot = ny;
       return;
    }
    Node denne = rot;
    boolean ferdig = false;
    while (!ferdig)
    {
       if (antall < denne.antall)
       {
         if (denne.venstre == null)
         {
            denne.venstre = ny;
            ferdig = true;
         }
         else
            denne = denne.venstre;
       }
       else
       {
         if (denne.høyre == null)
         {
            denne.høyre = ny;
            ferdig = true;
         }
         else
            denne = denne.høyre;
       }
    }
}
```

b)
```
void skriv(Node rot)
{
    if (rot != null)
    {
       skriv(rot.høyre);
       System.out.println(rot.antall + " " + rot.ord);
       skriv(rot.venstre);
    }
}
```

c)
```
void skriv(Node rot, int n)
{
   if (rot != null)
   {
       if (n < rot.antall)
          skriv(rot.venstre, n);
       else
       {
          if (n == rot.antall)
              System.out.println(rot.ord);
          skriv(rot.høyre, n);
       }
   }
}
```

Oppgave 9

a)
```
int hash(long fødselsnummer)
{
   return (int) fødselsnummer % hashtabell.length;
}
```

b)
```
public InnbyggerRegister(int maxAntall)
{
   hashtabell = new Innbygger[nestePrimtall(2 * maxAntall)];
}
```

c)
```
void settInn(long fødselsnummer, Innbygger I)
{
   int h = hash(fødselsnummer);
   while (hashtabell[h] != null)
   {
       h++;
       if (h == hashtabell.length)
        h = 0;
   }
   hashtabell[h] = I;
}
```