1) Labyrinten

Olle gör en djupet-förstsökning i labyrinten. För att implementera djupet-förstI behövs en stack för att hålla reda på tidigare vägval och en nodklass med höger- och vänsterpekare för att generera trädet.

2) Kökod

Endast det sist instoppade elementet behålls eftersom man inte förändrar någon next-pekare någonstans. Kön blir alltså:

’Annie’
’Bonnie’
’Clyde’

Koden fungerar bättre om man ändrar i else-satsen.

```
Node putInto(Object o, Node p) {
    if (p == null) return new Node(o);
    else {
        p.next = putInto(o, p.next);
        return p;
    }
}
```

Basfallet behöver inte ändras, om kön är tom stoppar jag in elementet och returnerar det. Annars stoppar jag in elementet i resten av listan och därefter returnerar jag min lista.

3) Heap

Hur man stoppar in i heap-vektorn står beskrivet i föreläsningsanteckningarna (9) eller kursboken.

```
133
133     520
133     520  800
13      133  800  520
13      87   800  520  133
13      87   800  520  133  900
```

4) Syntax för kanadensare

Alternativ 1 och 2 kan producera syntaxen.
Alternativ 3 och 4 kan inte producera *Båt 1 och båt 2 ska in!*

Alternativ 1 och 4 borde inte godkänna *Båt 4 och*
Alternativ 3 borde inte godkänna *Båt 1 och båt 2*

5) Kollisioner

Måste hantera krockar för att inte skriva över tidigare värden. Behövs inte med perfekta hashfunktioner – då blir det inga krockar. Visa krockhantering t.ex. krocklista. Klustring är stora klumpar i hashtabellen som måste genomsökas linjärt. Bloomfilter bryr sig inte om krockar utan löser det med flera hashfunktioner och sannolik träffsäkerhet.

Se föreläsning 10 för utförligare förklaring.

6) Simbassängen

Ta startbarnet ur kön, kom ihåg barnet (med en pekare eller spara namnet), pusha barnet på stacken.

Ta busungen ur kön, kom ihåg barnet (med en pekare eller spara namnet), pusha busungen på stacken.

Ta upprepade gånger nästa ur kön och pusha på stacken men om ett skötsamt barn kommer töm då stacken till kön.

Notera varje gång startbarnet eller första busungen passerar, bryt när startbarnet dyker upp tionde gången.

Skriv ut hur många gånger busungen passerat.

7) Dokusåpan

Urval eller bubbelsortering är OK, det blir (3 + 1) * 243 jämförelser. Observera att det behövs ett extra varv för att konstatera att kön är sorterad.

Smartare är att först sortera ut de som sticker ut (243 - 1) jämförelser
och sedan sortera dem 3 * log 3 = 3 jämförelser och sedan stoppa in dem först i kön => 245 jämförelser.

Det förekom andra lösningar som t.ex. räknesortering (antal näslängder är litet).

8) KMP

next-vektorn används när nästa tecken i söktexten inte matchar med nuvarande tillstånd i automaten. Värdet i next-vektorn talar om vilket tillstånd man ska kontrollera.

```
KANADAGÄSS KAN KANSKE KALLAS KANADENSISKA
01111111111011401140210131111011116111101
```