Exempeltenta för kursen Algoritmer och Datastrukturer för IoT-22.

2022-12-01

Tid	13:00 – 15:00
Hjälpmedel	Penna och valfritt antal tomma papper för anteckningar
Totalt	60 poäng (50p G + 10p VG)
Godkänt	30 poäng på G-delen
Väl godkänt	30 poäng på G-delen + 6 poäng på VG-delen

Instruktioner:

- Svar skall vara på svenska eller engelska. Answers must be in Swedish or English.
- Uppgifterna är inte ordnade efter svårighet.
- Programkod skall skrivas i Python.
- Programkod som finns given i uppgiften behöver inte repeteras.
- Givna deklarationer, parametrar, etc. får inte ändras, om inte annat sägs i uppgiften.
- Läs igenom alla frågor så du har gott om tid att ställa eventuella frågor till läraren om det skulle behövas.

1. **Enkel-länkad lista (10p)**
   Nedan kan du läsa kod som implementerar en enkellänkad lista; implementerad i klassen LinkedList.

   - Implementera koden för metoden **print_reverse()**
   - Metoden ska skriva ut den data som lagras i listans noder, med start på den sista noden.
   - Metoden ska skriva ut en nods data per rad.
   - Metoden ska implementeras med rekursion.

```
 1  class Node:
 2      def __init__(self, data):
 3          self.data = data
 4          self.next = None
 5
 6  class LinkedList:
 7      def __init__(self):
 8          self.head = None
 9
10      def print_reverse(self):
11          """ Print the linked list in reverse """
```

2. **Array (10p)**
   När en array roteras görs det genom att flytta ett element från ena änden till andra änden av arrayen.

   Implementera en funktion **rotate_left** som tar som input:
   - Ett heltal ***d***: Antal steg arrayen ska roteras
   - En lista ***arr***: Listan som ska roteras.

   Funktionen kan implementeras med loopar eller med rekursion.
   Funktionen ska returnera den roterade listan.

   Exempel:
   - [1, 2, 3, 4, 5] roterad vänster med 1 steg blir: [2, 3, 4, 5, 1].
   - [1, 2, 3, 4, 5] roterad vänster med 3 steg blir: [4, 5, 1, 2, 3]

```
 1  def rotate_left(d, arr):
 2      """ Rotate the list "arr" left, by "d" steps. """
```

3. Stack och Kö (10 p)

Nedan följer några påståenden om datatyperna Stack och Kö (Queue).
Välj ett svarsalternativ på varje fråga. Motivering krävs ej. 2 poäng per korrekt svar. Garderingar ger noll poäng.

3.1 Vad är sant om stack och kö?

a) Stack är FIFO och Kö är LIFO
b) Stack är LIFO och Kö är FIFO
c) Både Stack och Kö är FIFO
d) Både Stack och Kö är LIFO
e) Det beror på den underliggande datastrukturen

3.2 Vad är sant om komplexiteten av en Stack som är implementerad med en Enkel-Länkad Lista som underliggande datastruktur?

a) push(x) är O(1) och pop() är O(1)
b) push(x) är O(N) och pop() är O(N)
c) push(x) är O(1) och pop() är O(N)
d) push(x) är O(N) och pop() är O(1)

3.3 Vad är sant om komplexiteten av en Kö som är implementerad med en Array som underliggande datastruktur?
Enqueue(x) lägger till ett element i början av Kön, **Dequeue()** tar bort ett element i slutet av Kön.

a) enqueue(x) är O(1) och dequeue() är O(1)
b) enqueue (x) är O(N) och dequeue () är O(N)
c) enqueue (x) är O(1) och dequeue () är O(N)
d) enqueue (x) är O(N) och dequeue () är O(1)

3.4 Följande operationer körs på en stack. Vid startpunkten har stacken tre element: 1, 2, 3. Stacken växer åt höger när element läggs till.
pop(), push(4), peek(), push(3), pop(), is_empty(), push(4)

a) 1, 2, 3, 4
b) 2, 4, 3, 1
c) 4, 4, 2, 1
d) 1, 2, 4, 4
e) 4, 3, 2, 1

3.5 Följande operationer körs på en kö. Vid startpunkten har kön två element: 1, 2.
Nya element läggs till från vänster.
enqueue(3), enqueue(3), dequeue(), dequeue(), peek(), enqueue(1)

a) 3, 1, 1
b) 3, 2, 1
c) 1, 2 ,3
d) 1, 2, 1
e) 1, 3, 3

4. **Sortering – Quicksort (10p)**
   Givet arrayen i bilden:

24	20	19	26	31	25	21

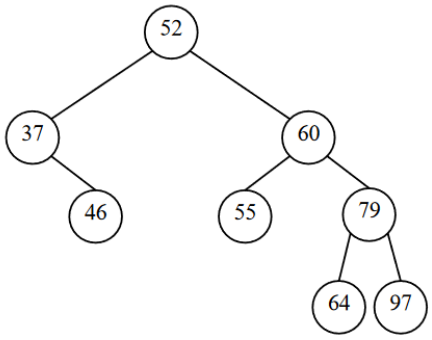
   Förklara steg för steg hur denna lista sorteras med hjälp av Quicksort-algoritmen.
   Antag att värdet längst till vänster används som pivot-element i varje steg.

5. **Sökning (10p)**
   Givet arrayen i bilden:

19	20	21	24	25	26	31

   Förklara steg för steg hur indexet för värdet 26 hittas med hjälp av binärsökning.
   Antag att det vänstra elementet alltid väljs när det inte finns något tydligt val av mitt-element.

6. **VG-uppgift: Binärt träd (10 poäng)**
   Givet det binära trädet i bilden nedan:



   Det binära trädet är ordnat så att alla värden är sorterade från minsta till största värde från vänster till höger.

   Uppgift:
   Förklara steg för steg hur en **rekursiv funktion** kan implementeras, som givet trädet ovan som input, skriver ut alla värden från det minsta värdet till det största värdet.

   Svar:
   Svaret kan anges som Python-kod, pseudo-kod eller beskrivet med dina egna ord.

   Exempel-output från funktionen:
   37
   46
   52
   55
   60
   64
   79
   97