ÉCOLE POLYTECHNIQUE MONTRÉAL

Le génie sans frontières

Questionnaire examen final

INF2010

Sigle du cours

Identification de l'étudiant(e)		
Nom :	**Prénom :**	
Signature :	**Matricule :**	**Groupe :**

Sigle et titre du cours	*Groupe*	*Trimestre*
INF2010 – Structures de données et algorithmes	**Tous**	**20083**
Professeur	***Local***	***Téléphone***
Ettore Merlo, responsable – Tarek Ould Bachir, chargé	**M-4105**	**5758 - 5193**

Jour	*Date*	*Durée*	*Heures*
Jeudi	**18 décembre 2008**	**2h30**	**9h30 – 12h00**

Documentation	*Calculatrice*	
☐ Toute ☒ Aucune ☐ Voir directives particulières	☐ Aucune ☐ Programmable ☒ Non programmable	**Les cellulaires, agendas électroniques ou téléavertisseurs sont interdits.**

Directives particulières

Bonne chance à tous!

Important

Cet examen contient **6** questions sur un total de **18** pages **(excluant cette page)**

La pondération de cet examen est de **40** **%**

Vous devez répondre sur : ☒ le questionnaire ☐ le cahier ☐ les deux

Vous devez remettre le questionnaire : ☒ oui ☐ non

L'étudiant doit honorer l'engagement pris lors de la signature du code de conduite.

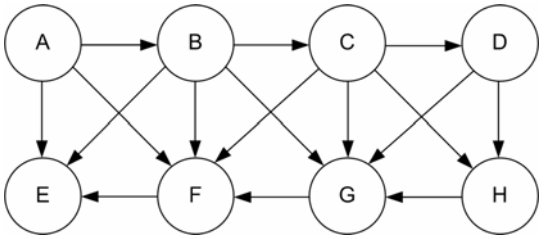
Question 1 : Graphes acycliques (15 points)

On vous présente un ensemble de graphes dirigés. Pour chacun des graphes qui suivent, indiquez si le graphe est cyclique.

1) Si c'est le cas, indiquez au moins un cycle.

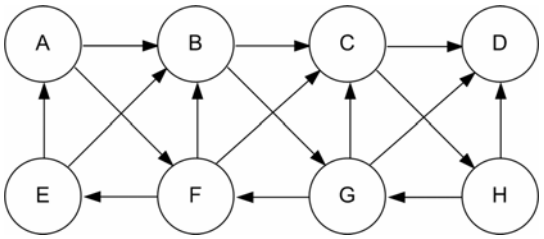
2) Sinon, numérotez les nœuds de **A** à **H** de sorte que, s'il existe un chemin du nœud **i** au nœud **j**, alors **i**<**j**.

a) **(5 pnt)** Cyclique : OUI ☐ NON ☐



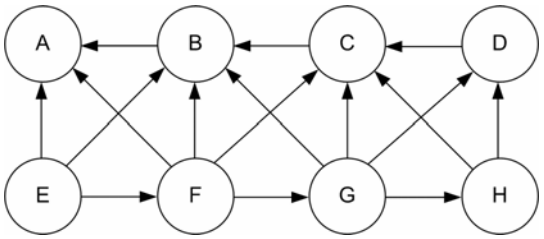
	Indegree avant la sortie de file							
Noeud	1	2	3	4	5	6	7	8
A								
B								
C								
D								
E								
F								
G								
H								
Entrée								
Sortie								

b) **(5 pnt)** Cyclique : OUI ☐ NON ☐



	Indegree avant la sortie de file							
Noeud	1	2	3	4	5	6	7	8
A								
B								
C								
D								
E								
F								
G								
H								
Entrée								
Sortie								

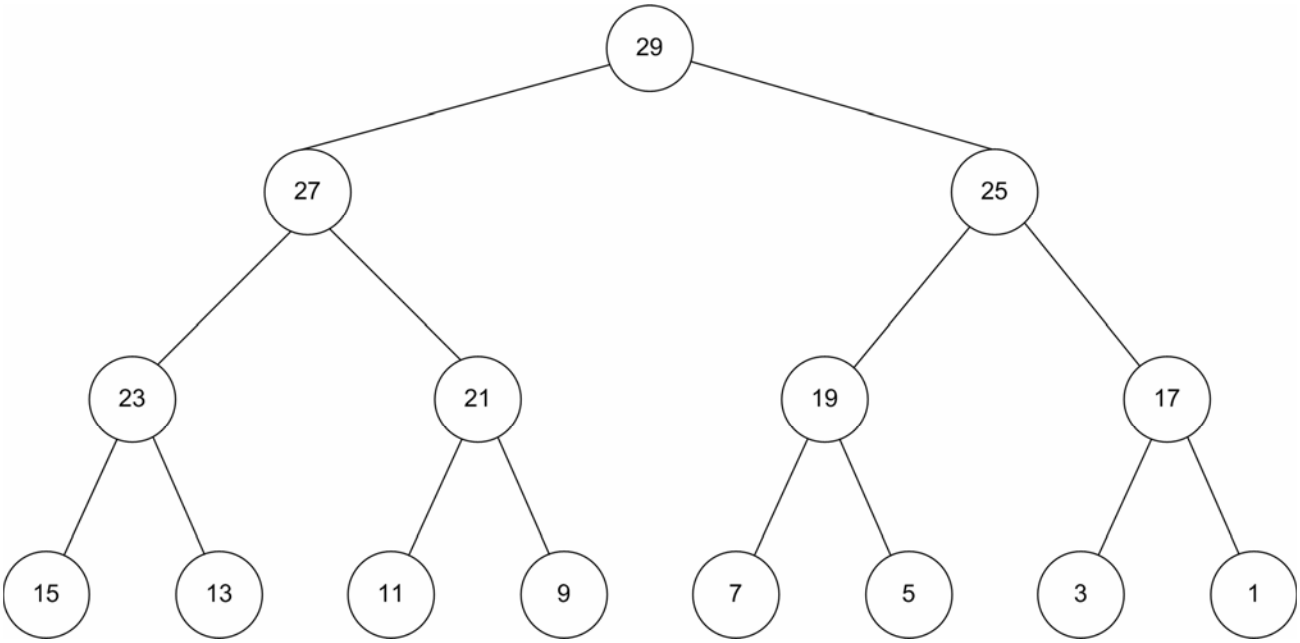
c) **(5 pnt)** Cyclique : OUI ☐ NON ☐



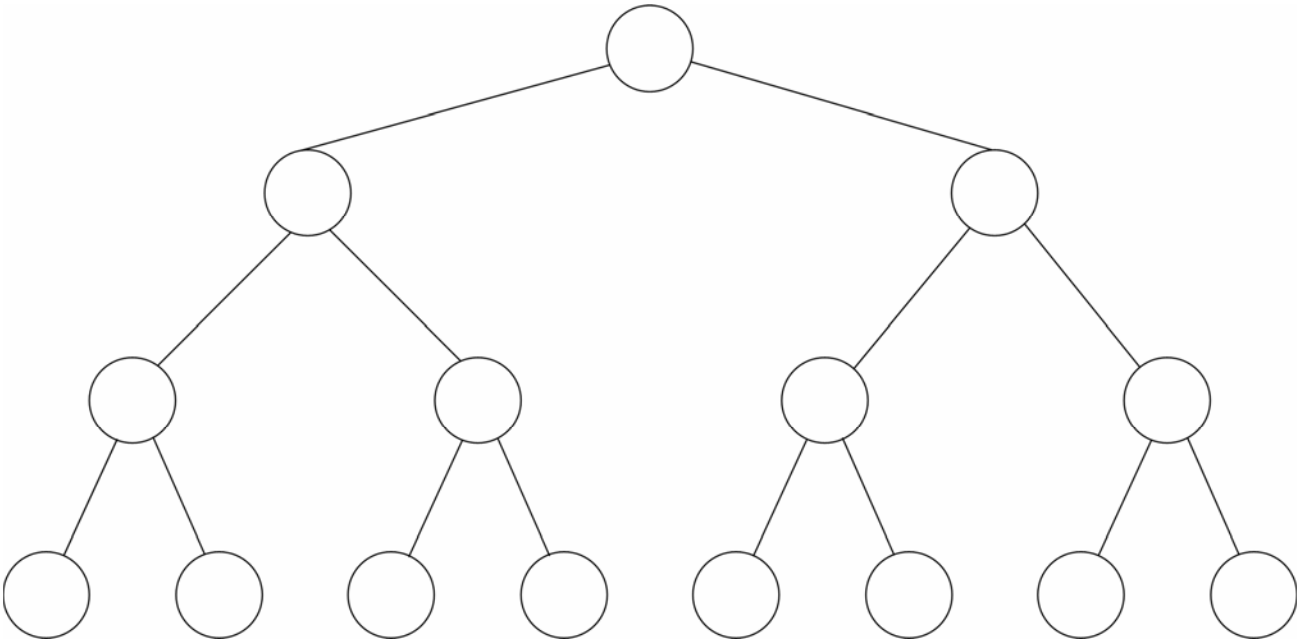
	Indegree avant la sortie de file							
Noeud	1	2	3	4	5	6	7	8
A								
B								
C								
D								
E								
F								
G								
H								
Entrée								
Sortie								

Question 2 : Monceaux (20 points)

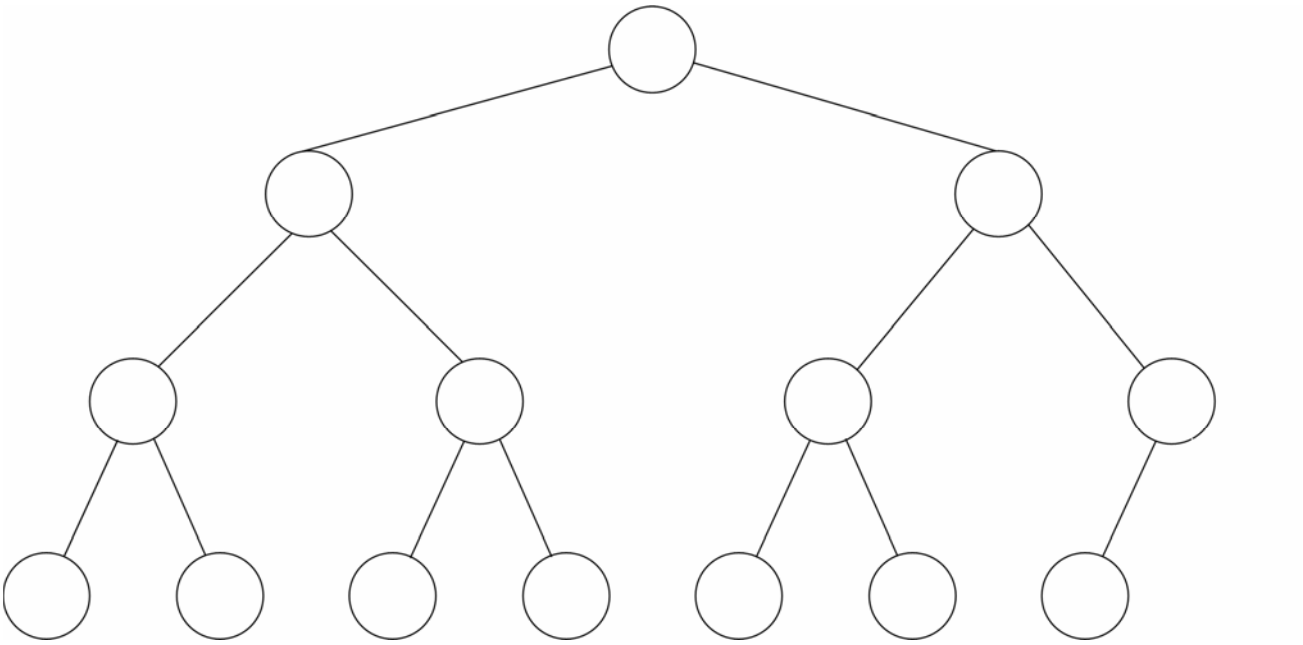
a) **(5 pnt)** Construisez, selon la technique vue dans le cours, un monceau à partir de l'arbre binaire suivant :



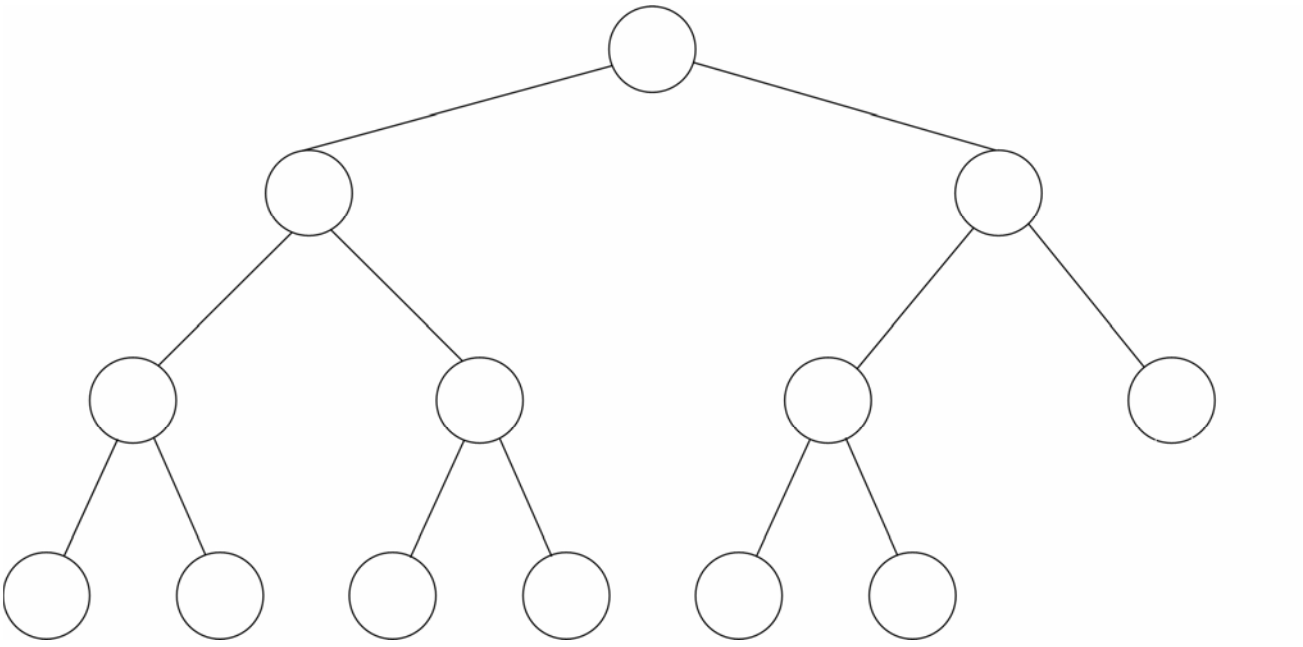
Monceau :



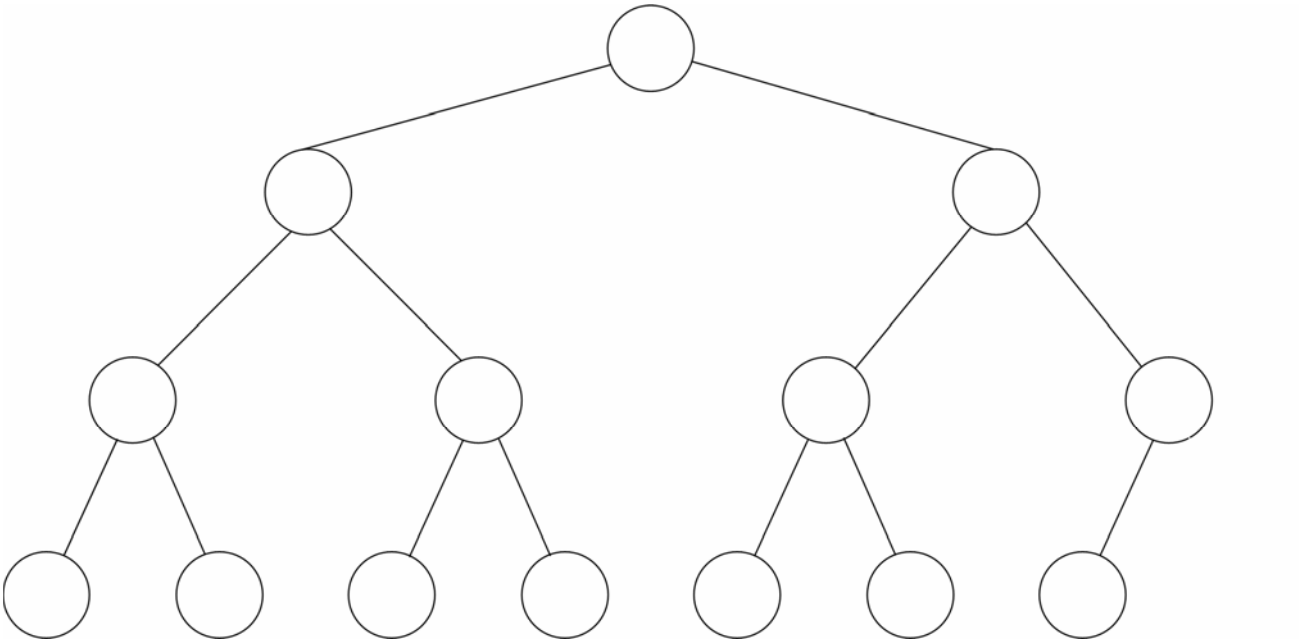
b) **(4 pnt)** Dessinez l'état du monceau ainsi obtenu après l'appel à `deleteMin()` :



c) **(4 pnt)** Dessinez l'état du monceau ainsi obtenu après l'appel à `deleteMin()` :



d) **(4 pnt)** Ajoutez au monceau ainsi obtenu un nœud dont la clé est 2 :

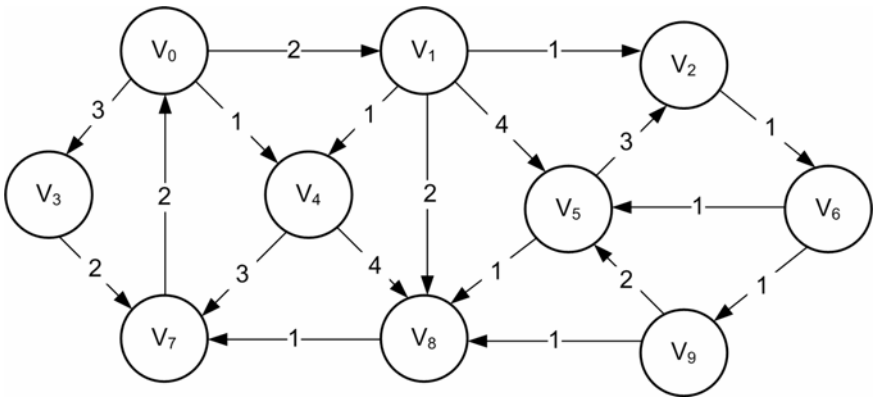


e) **(3 pnt)** Dessinez l'état du tableau contenant le monceau à la fin de ces opérations :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Question 3 : Le plus court chemin dans un graphe (15 points)

En appliquant l'algorithme de Dijkstra utilisant une file de priorité (ici est partiellement réalisé), trouvez la longueur du plus court chemin menant à chacun des nœuds du graphe en partant de V_0.



a) **(9 pnt)** Continuez l'exécution de l'algorithme en vous référant à l'état de la file de priorité.

Nœud	Connu	Dist min.	Parent
V_0	√	0	
V_1	√	~~∞~~, 2	V_0
V_2		~~∞~~, 3	V_1,
V_3		~~∞~~, 3	V_0
V_4	√	~~∞~~, 1	V_0
V_5		~~∞~~, 6	V_1,
V_6		∞	
V_7		~~∞~~, 4	V_4
V_8		~~∞~~, ~~5~~, 4	~~V_4~~, V_1
V_9		∞	

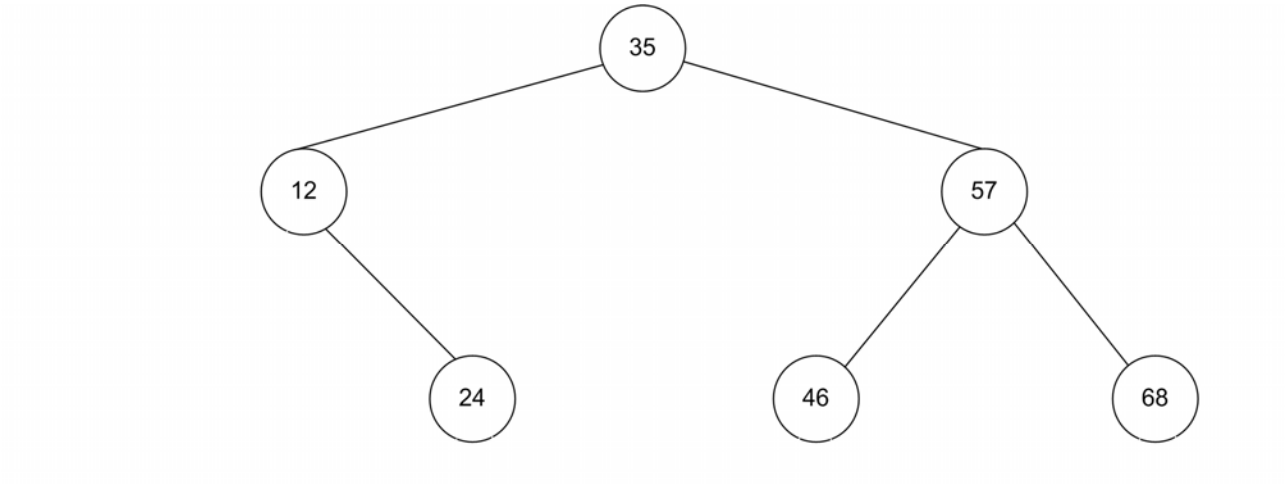
File de priorité
(V_3, 3)
(V_2, 3)
(V_7, 4)
(V_8, 4)
(V_5, 6)

b) **(6 pnt)** Détaillez chacun des chemins les plus courts trouvés :

Nœud	Le plus court chemin	Distance parcourue
V_1	V_0, _____, V_1	
V_2	V_0, _____, V_2	
V_3	V_0, _____, V_3	
V_4	V_0, _____, V_4	
V_5	V_0, _____, V_5	
V_6	V_0, _____, V_6	
V_7	V_0, _____, V_7	
V_8	V_0, _____, V_8	
V_9	V_0, _____, V_9	

Question 4 : AVL (15 points)

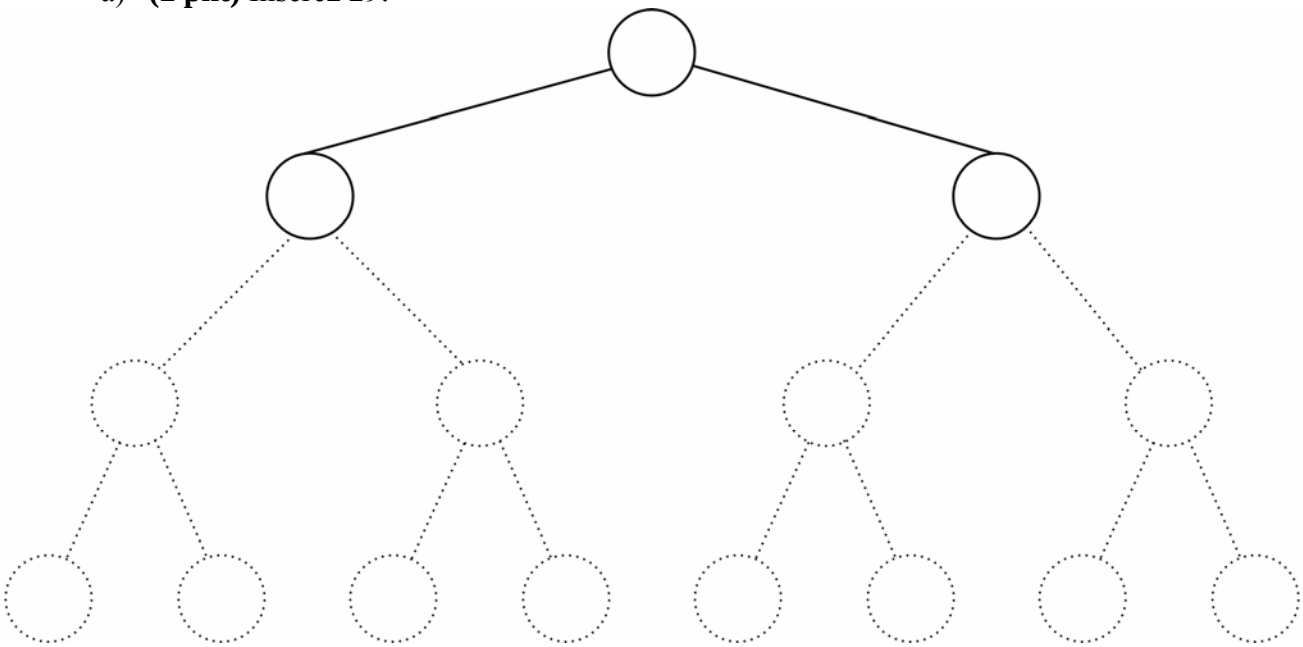
En considérant l'arbre AVL suivant :



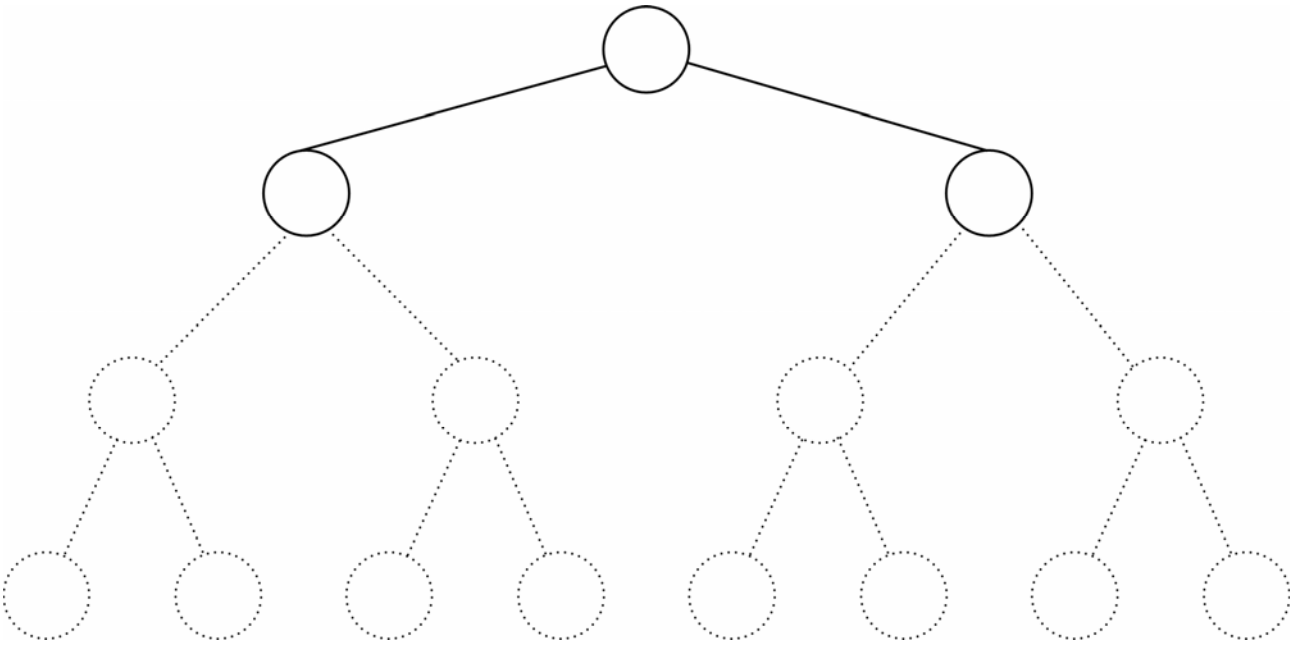
Effectuez l'ensemble des opérations suivantes dans l'ordre en vous servant des arbres ci-bas :

a) Insérez 29.
b) Insérez 32.
c) Insérez 51.
d) Insérez 34.
e) Insérez 33.
f) Insérez 49.

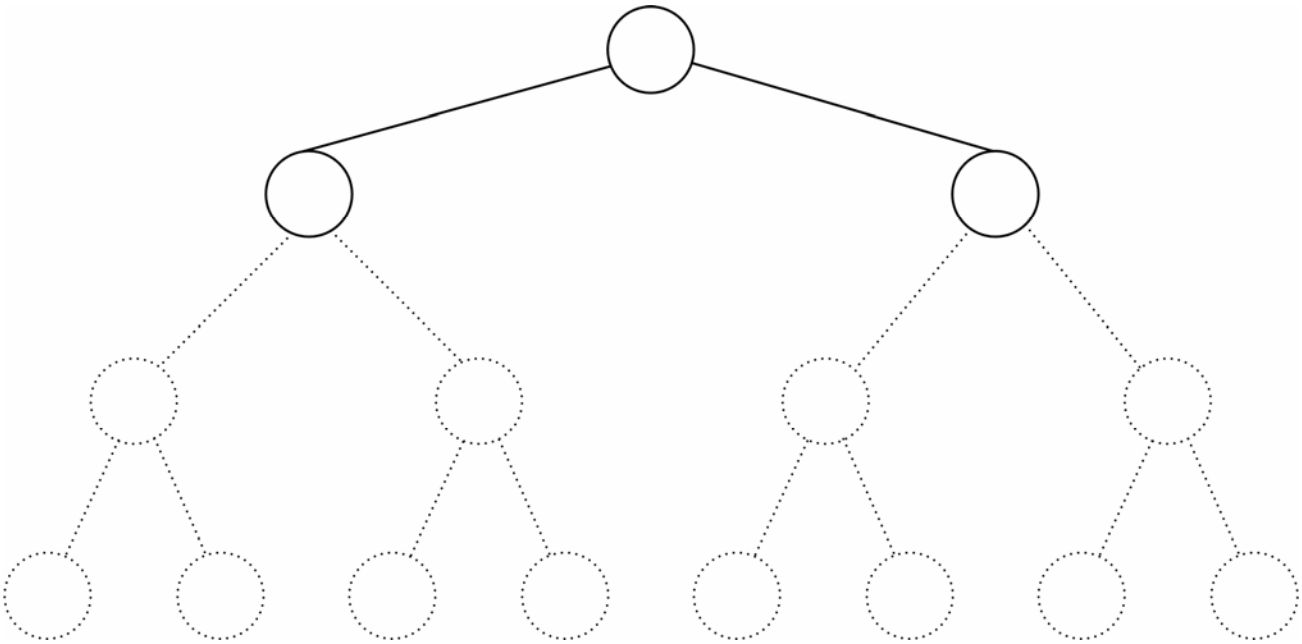
a) **(2 pnt)** Insérez 29.



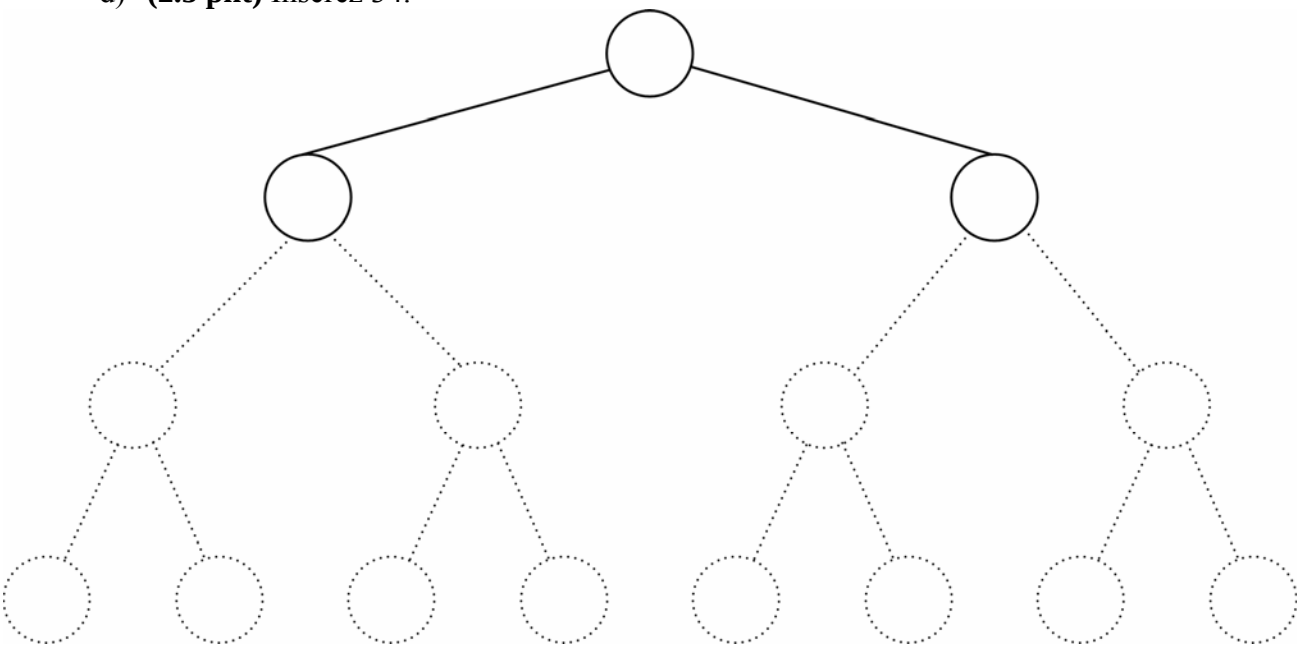
b) **(2 pnt)**Insérez 32.



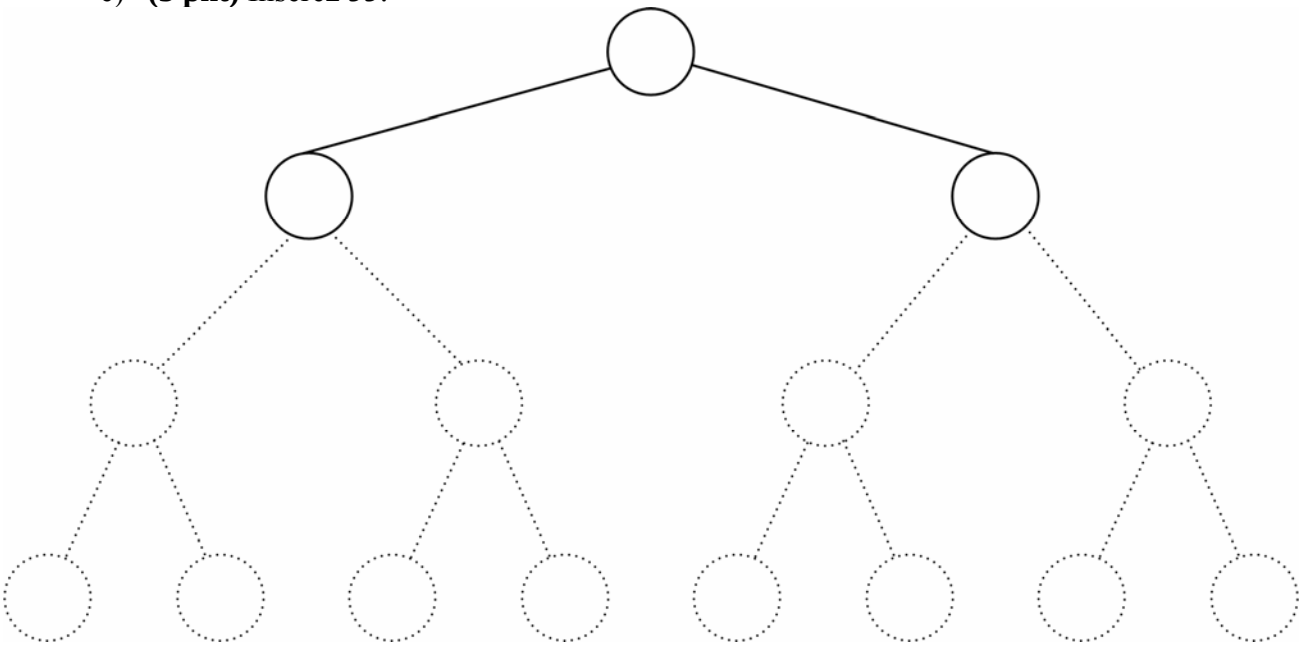
c) **(2.5 pnt)** Insérez 51.



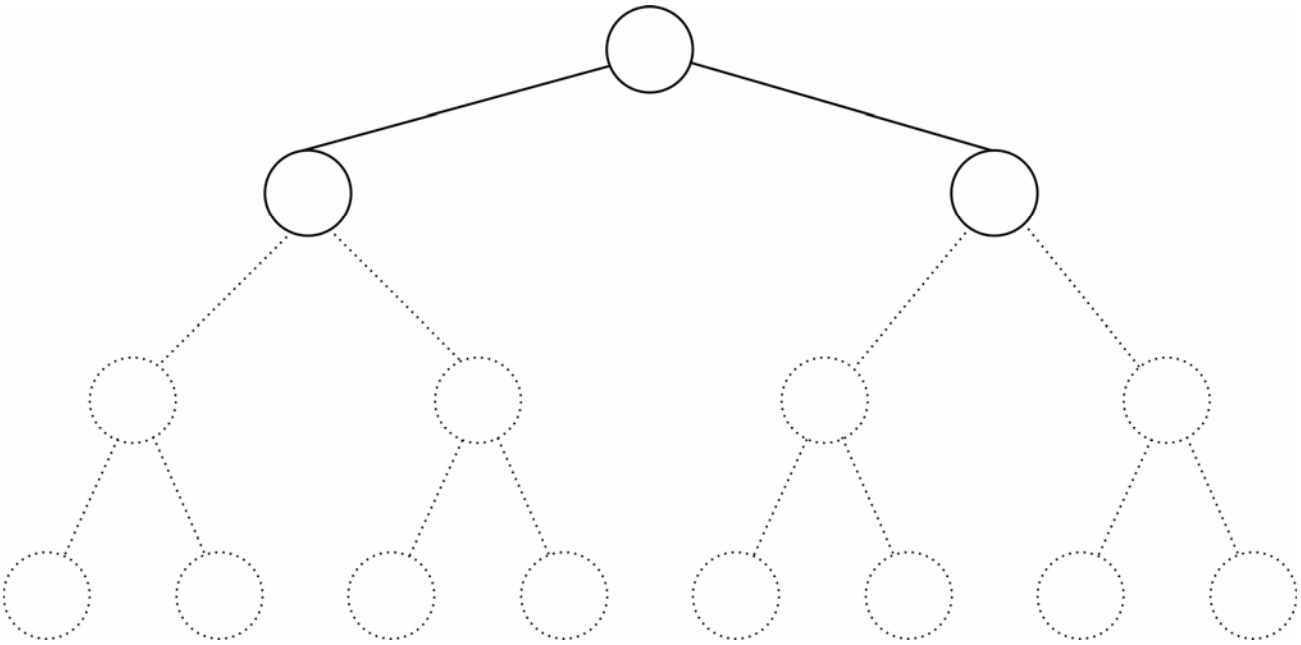
d) **(2.5 pnt)** Insérez 34.



e) **(3 pnt)** Insérez 33.



f) **(3 pnt)** Insérez 49.



Question 5 : Rabin-Karp (20 points)

L'algorithme Rabin Karp est un algorithme permettant de retrouver une chaîne de caractères dans un texte. La chaîne de caractères recherchée est alors remplacée par un nombre qu'il faut pré-calculer et on compare toutes les valeurs des sous-séquences du texte à la valeur pré-calculée. Les différentes implémentations (**code java**) discutées dans cette question vous sont données â l'**annexe I**.

Dans ces implémentations de Rabin-Karp, le texte et le patron ne contiennent que les caractères numériques '0'-'9' dans l'encodage ASCII. Les valeurs respectives de ces caractères sont données ci-dessous :

Caractère	Val. ASCII	Caractère	Val. ASCII
'0'	48	'5'	53
'1'	49	'6'	54
'2'	50	'7'	55
'3'	51	'8'	56
'4'	52	'9'	57

On vous propose trois méthodes pour calculer la valeur numérale associée aux sous-chaînes :

Méthode	Description
#1	Polynôme dans la base d = 58; ex : P = '123' p= (49*58+50)*58 + 51 = 167787
#2	Polynôme dans la base d = 10; ex : P = '123' p = (49*10+50)*58 +51 =5451
#3	Polynôme modulaire dans la base d = 58 modulo 10; ex : P= '123' p = 167787%10=7

a) **(5 pnt)** Donnez les avantages (au moins un) et les désavantages (au moins un) que vous voyez à utiliser chacune de ces méthodes :

Méthode	Avantages	Désavantages
#1		
#2		
#3		

On vous propose la méthode de calcul suivante (option par défaut dans le code) :

Polynôme modulaire dans la base d = 10 modulo 11

Par exemple, P = '123'

p = ((((49%11)*10 + 50) % 11) * 10 + 51) % 11
= ((((5)*10 + 50) % 11) * 10 + 51) % 11
= (((100) % 11) * 10 + 51) % 11
= ((1) * 10 + 51) % 11
= (61) % 11
= 6

b) **(3 pnt)** Donnez la valeur de p pour le patron P = '32123'

c) **(12 pnt)** Retrouvez tous les décalages donnant la présence de P dans le texte T= '3212323212321'

Décalage (s)	Sous-chaîne	Valeur du polynôme	Égalité?	Faux positif?	Correspondance?
0	'32123'				
1	'21232'				
2	'12323'				
3	'23232'				
4	'32321'				
5	'23212'				
6	'32123'				
7	'21232'				
8	'12321'				

Faux positifs trouvés :

Décalages retournés :

Question 6 : DP-Matching (15 points)

En utilisant le tableau suivant, retrouvez la plus longue sous-séquence commune aux chaînes d'entrée : X = 'CTGAATGACTAG' et Y = 'CATAGTCACTAG'

	Y	C	A	T	A	G	T	C	A	C	T	A	G
X	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0												
T	0												
G	0												
A	0												
A	0												
T	0												
G	0												
A	0												
C	0												
T	0												
A	0												
G	0												

Longueur de la plus longue sous-séquence commune :

Plus longue sous-séquence commune :

Annexe I

```
import java.util.ArrayList;


public class RabinKarp {

	public static ArrayList<Integer>
			RabinKarpFind(String Text, String Pattern)
	{
	  ArrayList<Integer> decalages = new ArrayList<Integer>();

	  if( Text.length() < Pattern.length() )
	    return decalages;

	  int p = ComputePatternValue( Pattern );

	  for(int i=0; i <= Text.length() - Pattern.length(); i++)
	  {
	    int t = ComputePatternValue(
	                Text.substring( i, i+Pattern.length() )
	                );

	    if( t == p)
	    {
	          int j;
	          for(j=0; j< Pattern.length(); j++)
	          {
	                  if( Pattern.charAt( j ) != Text.charAt( i +j ) )
	                  {
	                          break;
	                  }
	          }

	          if(j == Pattern.length())
	          {
	                  decalages.add( i );
	                  System.out.println("Correspodance à " + i);
	          }
	                  else
	          {
	                  System.out.println("Faux positif à " + i);
	          }

	    }

	  }

	  return decalages;
	}

	public static int ComputePatternValue(String Pattern)
	{
	  return ComputePatternValue(Pattern, 0);
	}
```

```
public static int ComputePatternValue(String Pattern, int method)
{
  // TODO Auto-generated method stub
  int p = 0;

  switch( method )
  {
  case 1:
   for(int i=0; i<Pattern.length(); i++)
   {
         p *=58;
         p += (int) Pattern.charAt( i );
   }
  break;

  case 2:
   for(int i=0; i<Pattern.length(); i++)
   {
         p *=10;
         p += (int) Pattern.charAt( i );
   }
  break;

  case 3:
   for(int i=0; i<Pattern.length(); i++)
   {
         p *=58;
         p += (int) Pattern.charAt( i );
         p %= 10;
   }
  break;

  default:
   for(int i=0; i<Pattern.length(); i++)
   {
         p *=10;
         p += (int) Pattern.charAt( i );
         p %= 11;
   }

  }

  return p;
}

public static void main(String[] args)
{
  // TODO Auto-generated method stub
  String Pattern = "123";
  int p = ComputePatternValue(Pattern, 1);
  System.out.println( p );

  p = ComputePatternValue(Pattern, 2);
  System.out.println( p );

  p = ComputePatternValue(Pattern, 3);
  System.out.println( p );
```

```
        p = ComputePatternValue(Pattern);
        System.out.println( p );

        Pattern = "32123";
        p = ComputePatternValue(Pattern);
        System.out.println( p );

        String Text = "3212323212321";
        ArrayList<Integer> decalages = RabinKarpFind(Text, Pattern);

        for(int s : decalages )
        {
         System.out.print( s + "\t" );
        }
      }
}
```