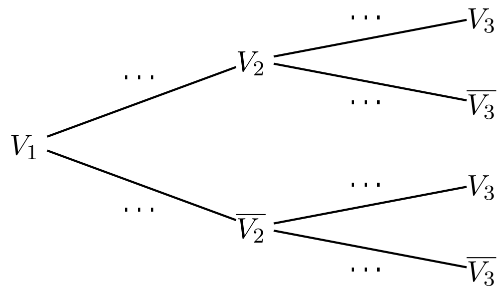
Dans tout l'exercice, les probabilités seront, si nécessaire, arrondies à 10^{-3} près.
Une donnée binaire est une donnée qui ne peut prendre que deux valeurs : 0 ou 1.
Une donnée de ce type est transmise successivement d'une machine à une autre.
Chaque machine transmet la donnée reçue soit de manière fidèle, c'est-à-dire en transmettant l'information telle qu'elle l'a reçue (1 devient 1 et 0 devient 0), soit de façon contraire (1 devient 0 et 0 et devient 1).
La transmission est fidèle dans 90 % des cas, et donc contraire dans 10 % des cas.
Dans tout l'exercice, la première machine reçoit toujours la valeur 1.

Partie A

Pour tout entier naturel $n \geqslant 1$, on note :

- V_n l'évènement : « la n-ième machine détient la valeur 1 » ;
- $\overline{V_n}$ l'évènement : « la n-ième machine détient la valeur 0 ».

1. (a) Recopier et compléter l'arbre de probabilité ci-dessous.



   (b) Démontrer que $P(V_3) = 0,82$ et interpréter ce résultat dans le contexte de l'exercice.
   (c) Sachant que la troisième machine a reçu la valeur 1, calculer la probabilité que la deuxième machine ait aussi reçu la valeur 1.

2. Pour tout entier naturel $n \geqslant 1$, on note $p_n = P(V_n)$.
   La première machine a reçu la valeur 1, on a donc $p_1 = 1$.
   (a) Démontrer que pour tout entier naturel $n \geqslant 1$:

$$p_{n+1} = 0,8p_n + 0,1$$

   (b) Démontrer par récurrence que pour tout entier naturel $n \geqslant 1$,

$$p_n = 0,5 \times 0,8^{n-1} + 0,5$$

   (c) Calculer la limite de p_n lorsque n tend vers l'infini. Interpréter ce résultat dans le contexte de l'exercice.

Partie B

Pour modéliser en langage Python la transmission de la donnée binaire décrite en début d'exercice, on considère la fonction `simulation` qui prend en paramètre un entier naturel n qui représente le nombre de transmissions réalisées d'une machine à une autre, et qui renvoie la liste des valeurs successives de la donnée binaire.
On donne ci-dessous le script incomplet de cette fonction.
On rappelle que l'instruction `rand()` renvoie un nombre aléatoire de l'intervalle $[0\ ;\ 1[$.

```
1   def simulation(n):
2      donnee = 1
3      liste = [donnee]
4      for k in range(n):
5          if rand() <0.1
6             donnee = 1 - donnee
7          liste.append(donnee)
9      return liste
```

Par exemple, `simulation(3)` peut renvoyer $[1,\ 0,\ 0,\ 1]$. Cette liste traduit :

- qu'une donnée binaire a été successivement transmise trois fois entre quatre machines ;
- la première machine qui détient la valeur 1 a transmis de façon contraire cette donnée à la deuxième machine ;
- la deuxième machine a transmis la donnée qu'elle détient de façon fidèle à la troisième ;
- la troisième machine a transmis de façon contraire la donnée qu'elle détient à la quatrième.

1. Déterminer le rôle des instructions des lignes 5 et 6 de l'algorithme ci-dessus.
2. Calculer la probabilité que `simulation(4)` renvoie la liste $[1,\ 1,\ 1,\ 1,\ 1]$ et la probabilité que `simulation(6)` renvoie la liste $[1,\ 0,\ 1,\ 0,\ 0,\ 1,\ 1]$.