Le « jeu de la boule » ou « petite roulette » est un jeu de casino similaire à celui de la roulette, mais avec les chiffres de 1 à 9.

*Il est possible de miser sur une **chance simple** (rouge, noir, manque, passe, pair ou impair) ou un **numéro plein**.*

*Les chiffres 1, 3, 7, 9 sont impairs. Les chiffres 2, 4, 6, 8 sont pairs. Les chiffres 1, 3, 6, 8 sont noirs. Les chiffres 2, 4, 7, 9 sont rouges. Les chiffres 1, 2, 3, 4 sont manques. **Les chiffres 6, 7, 8, 9 sont passes**. Le chiffre 5 n'est ni pair, ni impair, ni manque, ni passe, ni rouge, ni noir. **Si le 5 sort, la mise jouée** sur une chance simple **est perdue**.*

*Si la **chance simple** misée sort, le joueur gagne **une fois la mise**, sinon la mise est perdue.*

*Si le **numéro** misé sort, le joueur gagne **7 fois la mise** sinon la mise est perdue.*

PARTIE 1 - UNE PETITE PARTIE.

On souhaite écrire un algorithme qui simule UNE partie de jeu de la petite roulette, en misant sur « passe » et affiche à la fin la mise et le gain.

La variable X sera le numéro qui sort.

La variable M sera la mise, qu'on initialisera à 1.

La variable G sera le gain qu'on initialisera à 0.

→ Compléter l'algorithme ci-contre

```
Algorithme « ROULETTE »
 M prend la valeur ……
 G prend la valeur ……
 X prend la valeur d'un nombre entier
 aléatoire entre 1 et 9
 Si X > ……
 Alors G prend la valeur ……
```

PARTIE 2 - LA MARTINGALE INFAILLIBLE ?

Un joueur pense avoir trouvé une **martingale** infaillible :

– On mise 1 € : soit on gagne 2 € moins notre mise précédente 2 – 1 = 1 €, soit on perd et…

– … on mise 2 € : soit on gagne 4 € moins notre mise précédente : 4 – 3 = 1 €, soit on perd et…

– … on mise 4 € : soit on gagne 8 € moins notre mise précédente 8 – 4 – 2 – 1 = 1 €, soit on perd et…

– … on mise 8 € : soit on gagne 16 € moins notre mise précédente 16 – 8 – 4 – 2 – 1 = 1 €, soit on perd et…

Pour simuler cette martingale, on va améliorer notre algorithme, pour lequel on aura besoin de nouvelles variables :

La variable P, qui compte le nombre de parties, qu'on initialisera à 0

Les variable S et T qui cumuleront les Mises (S) et Gains (T) successifs qu'on initialisera à 0 en début d'algorithme.

```
Algorithme « ROULETTE »
 P prend la valeur 0
 S prend la valeur 0
 T prend la valeur 0
              [Partie 4]
 M prend la valeur 1
 G prend la valeur 0
  Tant que G = 0
              [Partie 3]
    S prend la valeur S + M
    P prend la valeur P + 1
    X prend la valeur d'un nombre
    entier aléatoire entre 1 et 9
    Si X > 5
     Alors G prend la valeur 2*M
     Sinon M prend la valeur 2*M
    Fin de boucle Si
    T prend la valeur T + G
  Fin de boucle Tant que
              [Partie 4]
 Afficher « Parties » puis P
 Afficher « Mise totale » puis S
 Afficher « Gain total » puis T
```

→ Quelle(s) instruction(s) de l'algorithme traduit …

A. Si on perd, on double la mise dans la partie suivante.	**B.** On joue jusqu'à ce qu'on gagne.	**C.** On gagne et on empoche le double de notre mise	**D.** On joue une partie de plus.

PARTIE 3 - GARE AU PLAFOND

Attention : Pour éviter des mises astronomiques que le casino ne serait pas capable de payer, la table de jeu est **plafonnée à 100 €**, on ne peut donc pas miser plus que cette valeur.

→ Rajouter à l'algorithme précédent la (les) instruction(s) nécessaire(s) pour tenir compte de ce paramètre.

PARTIE 4 - UNE SOIREE AU CASINO

On souhaite maintenant tester cette martingale sur le long terme. Au rythme d'une partie toutes les deux minutes, le joueur estime que dans la soirée il pourra jouer environ 150 parties.

→ Rajouter à l'algorithme précédent la (les) instruction(s) nécessaire(s) pour tenir compte de cette nouvelle contrainte (on rappelle que la variable P compte le nombre de parties).