Projets

1 Instructions générales

Ces projets se font en binôme. Chaque binôme doit choisir un projet parmi les projets 1 et 2 puis y résoudre le problème proposé. Le projet 3 est obligatoire. Il y a donc deux projets en tout à réaliser : (1 ou 2) et 3.

Le tout devra être rendu dans un dossier au format Nom1_Nom2 et les fichiers seront nommés selon le numéro du projet, par exemple, si vous avez choisi le projet 1, votre fichier se nommera projet1.py. Tout manquement à ces consignes sera sanctionné par un retrait de points.

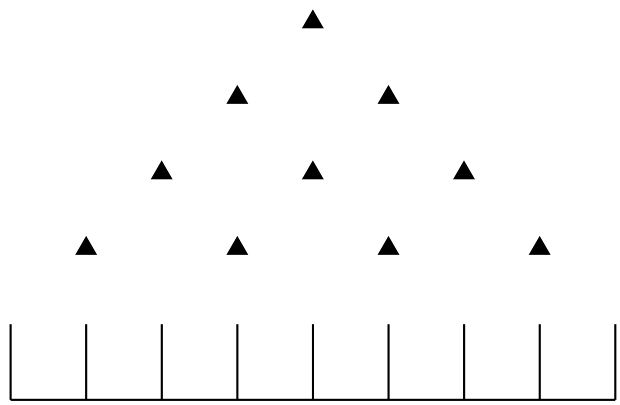
Les programmes rendus doivent être personnels et de votre niveau, tout programme trop proche de celui d'un autre élève, d'un autre trouvé sur internet ou d'un niveau supérieur à celui de première se verra attribué automatiquement la note 0 plus un rapport de fraude.

Les critères d'évaluations sont les suivants :

- la réalisation de l'objectif du projet (et donc le rendu d'un programme fonctionnel) ;
- l'efficacité du programme ;
- la clarté et la structure du programme, l'usage de commentaires ;
- l'usage adéquat de fonctions, boucles, tests, etc ;
- le nommage des variables.

2 Projet 1 : la planche de Galton

Une planche de Galton est une planche verticale sur laquelle sont disposés des « piquets » sous formes de pyramides comme sur le schéma ci-dessous avec une pyramide à quatre étages.



Les planches de Galton sont faites de façon à ce qu'une bille tombant sur l'un des piquets ait autant de chances de tomber à gauche qu'à droite. En tombant, elle va nécessairement sur un autre piquet jusqu'à temps qu'elle ait traversé toute la pyramide. Une fois la pyramide traversée, la bille tombe dans un des récipients situés en dessous. On peut alors compter le nombre de billes dans chaque récipient pour déterminer la répartition des billes.

1. Créer une fonction `plancheGalton()` simulant la chute d'une bille dans une planche de Galton de taille quelconque. *Indications :* faire le lien entre le nombre d'étages de la planche et le nombre de positions possibles de la bille puis raisonner en terme de position étape par étape.
2. Simuler la chute de 1000 billes dans la planche de Galton et enregistre le nombre de billes arrivant dans chacune des positions finales.
3. Afficher le résultat sous forme d'histogramme.
4. Reproduire plusieurs fois l'expérience (en faisant varier le nombre de billes ou la taille de la planche éventuellement). Qu'en déduisez-vous ?

3 Projet 2 : impôt sur le revenu

L'impôt sur le revenu se calcule à partir des revenus annuels d'une personne avec un fonctionnement par tranches. En 2021, les tranches et taux d'impositions pour celles-ci étaient donnés par le tableau ci-dessous.

Tranches	$[0\,;10064[$	$[10064\,;25660[$	$[25660\,;73370[$	$[73370\,;157807[$	$[157807\,;+\infty[$
Taux d'imposition	0%	11%	30%	41%	45%

Exemples :

— Pour un salaire annuel de 10000€. Le salaire est inférieur à 10064€ donc on est dans la première tranche ; on est non imposable.

— Pour un salaire annuel de 20000€. L'erreur est de considérer que l'on est taxé à 11%. Le salaire est en fait découpé selon les tranches :

Tranches	$[0\,;10064[$	$[10064\,;25660[$
Calcul du montant dans la tranche	10064-0=10064	20000-10064=9036
Taux d'imposition	0%	11%
Montant prélevé sur la tranche	$10064 \times 0\% = 0$	$9036 \times 11\% = 993,96$

Pour un salaire annuel de 20000€, on doit donc payer 993,96€. Cela représente 4,96% du salaire annuel et non 11%.

— Un salaire annuel de 200000€ sera découpé ainsi :

$$\begin{aligned}200000 &= (10064-0)+(25660-10064)+(73370-25660)+(157807-73370)\\&\quad+(200000-157807)\\&= \underbrace{10064}_{\text{tranche 1}} + \underbrace{15596}_{\text{tranche 2}} + \underbrace{47710}_{\text{tranche 3}} + \underbrace{84437}_{\text{tranche 4}} + \underbrace{42193}_{\text{tranche 5}}.\end{aligned}$$

Le montant à prélever est alors calculé comme suit :

$$10064 \times 0\% + 15596 \times 11\% + 47710 \times 30\% + 84437 \times 41\% + 42193 \times 45\% = 69634.58,$$

soit un taux d'imposition réel de 34.82%.

1. Écrire un programme permettant de calculer le montant des impôts sur le revenu, le taux d'imposition réel associé et les revenus restants après imposition pour un salaire annuel quelconque, des tranches d'impositions quelconques et des taux d'impositions par tranches quelconques.
2. Sur deux graphiques différents, afficher pour des salaires annuels allant de 1000€ à 1000000€ en augmentant de 1000 en 1000 :
   (a) le montant prélevé et le montant restant après prélèvement ;
   (b) le taux d'imposition réel.
3. Reproduire l'expérience en changeant (à votre convenance) les tranches (nombres et bornes) et leurs taux d'impositions. On pourra notamment expérimenter la dernière tranche à 0% et 100%.

4 Projet 3 : rendu de monnaie

4.1 Algorithme glouton

Un algorithme **glouton** est souvent une solution intéressante pour un problème d'optimisation. Ces problèmes ont deux caractéristiques : une fonction que l'on doit maximiser ou minimiser et une série de contraintes auxquelles il faut satisfaire. Dans les algorithmes gloutons, on fait toujours le choix qui semble meilleur sur le moment. Autrement dit, on fait un choix optimal localement dans l'espoir que ce choix mènera à la solution optimale globalement. Les algorithmes gloutons n'aboutissent pas toujours à des solutions optimales, mais ils y arrivent dans de nombreux cas.

4.2 Le rendu de monnaie

L'un des grands classiques des algorithmes gloutons est le problème du rendu de monnaie. En travaillant sur les euros, la stratégie gloutonne donne toujours une solution optimale au problème du rendu de monnaie.

Considérons le problème d'un commerçant devant rendre de la monnaie à l'un de ses clients. Il souhaite le faire en utilisant le moins de pièces et de billets possibles. On suppose que l'on manipule les coupures et pièces habituelles des euros (oublions les centimes) et que le commerçant dispose d'une réserve suffisamment importante de chaque espèce.

Si la somme qui doit être rendue est 9, les différentes combinaisons possibles sont les suivantes :

Combinaison	Pièces
$9 \times 1€$	9
$7 \times 1€ + 1 \times 2€$	8
$5 \times 1€ + 2 \times 2€$	7
$3 \times 1€ + 3 \times 2€$	6
$1 \times 1€ + 4 \times 2€$	5
$4 \times 1€ + 1 \times 5€$	5
$2 \times 1€ + 1 \times 2€ + 1 \times 5€$	4
$2 \times 2€ + 1 \times 5€$	3

La meilleure solution est donc de rendre 1 billet de 5€ et 2 pièces de 2€.

Pour aborder le problème de rendu de monnaie avec une stratégie gloutonne, on va donc sélectionner les pièces et billets à rendre un à un, et faire décroître progressivement la somme restant à rendre. Chaque choix doit être celui qui parait le meilleur au vu de la situation présente, c'est-à-dire de la somme restant à rendre. Pour limiter le nombre de devises rendues, on choisit de faire décroître cette somme aussi vite que possible.

Votre mission, si vous l'acceptez est de programmer une fonction prenant en entrée une somme à rendre et donnant en sortie le nombre de pièces et de billets correspondants avec une approche gloutonne.