Liste des projets de fin d'année 1 ère NSI 104 Jean Jaurès 2021.

Ce projet de fin d'année est un peu plus ambitieux que les précédents (analyse de vocabulaire, et site). Il sera présenté à l'oral devant la classe, et fera l'objet d'un (court) rapport.

- Les projets se font par groupe de deux. Pour les projets plus difficiles **(*) ou +, il est possible de faire le projet à 3. Les indications de difficulté sont indicatives.
- La majorité des projets a une partie « métier », où tous les calculs sont effectués, et une partie « graphique », d'interface avec l'utilisateur. Il est indispensable de bien séparer ces deux parties, même si, pour la mise au point, on peut mettre des « print » où l'on veut (et les supprimer ensuite). A deux, il est fort possible de séparer presque entièrement le travail. La partie graphique est en général plus facile à faire, même s'il n'est pas facile de rentrer dans la logique de tkinter.
- En Python, l'interface graphique interface se fait avec la bibliothèque tkinter. Avant de se lancer dans le projet, faire le notebook « interfaces graphiques », afin de comprendre les bases du fonctionnement de cette bibliothèque.
- Il est possible de faire certains de ces projets en version web (html/JavaScript). Vu le peu de temps que l'on a passé sur JavaScript, cela sera plus difficile.
- Tous les jeux de type « arcade » posent des problèmes de gestion du temps qui ne sont pas toujours facile à résoudre.
- Certains projets sont l'occasion de découvrir la programmation objet. Cela peut être intéressant pour ceux d'entre vous qui ont un bon niveau, et vont abandonner la spécialité. Pour ceux qui conservent la spécialité, la programmation objet sera vue en terminale.
- Vous pouvez proposer d'autres sujets. Essayez d'en évaluer la difficulté avant, ni trop peu, ni trop (ce qui est fréquent). Par exemple, le Pac-Man est faisable mais comporte en fait deux projets en un seul.
- Jeux de type shooters interdits, ça n'a aucun intérêt du point de vue programmation.
- Bibliothèque Pygame fortement déconseillée. Il est nécessaire d'en vérifier la compatibilité avec les dernières versions de Python, ce qui arrive rarement. Par ailleurs programmer avec Pygame semble vous encourager à tout mélanger entre métier et graphique, ce qui est une très mauvaise habitude.
- La présentation à l'oral se fait devant la classe, en 5-10 minutes.
- Le rapport très succinct doit comporter :
    - la répartition des tâches ;
    - le journal de bord/suivi du projet ;
    - les éventuelles astuces utilisées pour répondre au sujet ;
    - les algorithmes en langage naturel des points difficiles du sujet. On peut se permettre dans ces algorithmes des raccourcis du type « trier par ordre décroissant suivant *n* puis par ordre croissant suivant *m* ».

Exemple de progression version 1 :
Étape 1 : étude du cahier des charges, imaginer le programme (lister tout ce qu'il doit faire)
Étape 2 : déterminer les différentes fonctions de la partie métier/partie graphique, répartition des tâches dans l'équipe. Il est fortement déconseillé de programmer avant d'avoir fait les étapes des étapes 1 et 2.
Étapes 3 à 6 : version fonctionnelle de la partie métier.
Étapes 4 à 7 : version fonctionnelle avec la partie graphique. Suivant la répartition des tâches, se fait entièrement en parallèle ou bien un peu décalé de la partie métier.
Étape 8 : correction des derniers petits bugs, ajout des derniers commentaires.
Étape 9 : préparation de la présentation orale et rédaction du mini rapport
Exemple de progression version 2 :
Idem pour les étapes 1,2 8 et 9. Pour les autres étapes, la partie algo est faite entièrement avant la partie graphique

1. Le Jeu de la vie *
   Joli sujet. Peut se faire en programmation objet
   Extrait de WIKIPEDIA http://fr.wikipedia.org/wiki/Jeu_de_la_vie

En préambule, il faut préciser que le jeu de la vie n'est pas vraiment un jeu au sens ludique, puisqu'il ne nécessite aucun joueur ; il s'agit d'un automate cellulaire, un modèle où chaque état conduit mécaniquement à l'état suivant à partir de règles préétablies.
Le jeu se déroule sur une grille à deux dimensions, théoriquement infinie (mais de longueur et de largeur finies et plus ou moins grandes dans la pratique), dont les cases — qu'on appelle des « cellules », par analogie avec les cellules vivantes — peuvent prendre deux états distincts : « vivantes » ou « mortes ».
À chaque étape, l'évolution d'une cellule est entièrement déterminée par l'état de ses huit voisines de la façon suivante :

- Une cellule morte possédant exactement trois voisines vivantes devient vivante (elle naît)
- Une cellule vivante possédant deux ou trois voisines vivantes le reste, sinon elle meurt

Pour aller plus loin

- Faire fonctionner une grille à la façon d'un tore (on peut penser à une chambre à air comme représentation d'un tore)
    - Tout ce qui disparaît par le bas réapparaît par le haut et inversement
    - Tout ce qui disparaît par la gauche réapparaît par la droite et inversement

2. Le compte est bon **(*)

o Sujet avec une interface graphique minimale. Sujet pour ceux qui aiment réfléchir, les algorithmes de résolutions sont de ceux qui apportent le plus au niveau programmation, parmi tous les sujets proposés ici. Peut permettre un approfondissement sur la récursivité. Comme pour la programmation objet, la récursivité sera vue en terminale, mais est intéressante pour ceux qui abandonnent la spécialité. On peut utiliser un algorithme trouvé sur internet pour résoudre le problème arithmétique

Règles du jeu

- La donne :
    - Il y a 24 plaques comportant chacune un nombre entier positif
        - Deux plaques pour chaque entier de l'ensemble {1, 2, 3, 4, 5, 6, 7, 8, 9, 10}
        - Une plaque pour chaque entier de l'ensemble {25, 50, 75, 100}
        - On tire aléatoirement, avec équiprobabilité et sans remise, 6 des 28 plaques
        - On tire aléatoirement, avec équiprobabilité, un entier n dans l'intervalle [100 ; 999]
- L'objectif :
    - Obtenir n comme résultat d'une suite d'opérations arithmétiques, ou sinon approcher n au plus près, en combinant certaines des 6 plaques tirées et au moyen des 4 opérations +, - , × , /
- Règles à observer :
    - On ne peut utiliser une plaque qu'une fois, mais on n'est pas obligé de les utiliser toutes
    - On ne peut utiliser la division , que lorsqu'elle "tombe juste", et le résultat à prendre en compte est le quotient
    - Les entiers négatifs sont interdits

Pour aller plus loin

- Concevoir un algorithme résolvant le problème en utilisant le plus petit nombre de plaques possible
- Donner toutes les solutions possibles

3. Le tic-tac-toe *(*). Variante puissance 4

Sujet très classique. L'intelligence artificielle est obligatoire, elle est plus dure à faire pour le puissance 4 (pas de stratégie gagnante à coup sûr me semble-t-il).

Règles du jeu

- Deux joueurs impriment à tour de rôle leur marque, en général un rond pour l'un et une croix pour l'autre, dans les cases au départ vierges d'un plateau de 3 lignes et 3 colonnes (respectivement 4 pour le puissance 4)
- A gagné le premier joueur qui réussit à aligner 3 de ses marques, en ligne, en colonne ou en diagonale (respectivement 4)
- Le jeu peut se terminer sur un match nul

4. Bataille navale **

Sujet très classique. Moins simple qu'il n'y paraît. Le nombre de bateaux est à votre choix, par contre deux bateaux ne doivent pas se toucher. Le programme fait jouer un humain contre un ordinateur (idiot)

Règles du jeu

- Classiques, à choisir parmi les diverses variantes usuelles, voir sur internet si vous n'avez jamais joué à la bataille navale ☺

Pour aller plus loin

- Faire jouer une partie humain contre ordinateur intelligent

5. Générateur d'exercices de calcul mental *.

Présentation du problème.

Le but est de créer un programme permettant de tester les capacités en calcul mental d'un utilisateur, sur des entiers naturels. La difficulté est donnée par la taille des entiers et la nature des opérations. Sujet également intéressant en version web (avec du PHP et du JavaScript)

L'interface doit avoir :

- Choix de la difficulté (ou absence de choix)
- Choix de la version : QCM ou non.
- Affichage des questions, des réponses et du score final

Pour aller plus loin

- Dans la version QCM, donner la réponse juste et trois réponses plausibles.

6. Calcul en valeur exacte sur des fractions *

La partie « pour aller plus loin » est la bienvenue, sinon le projet est un peu juste. Le professeur vous expliquera ce que sont les fractions continues.

Présentation du problème.

Le but est de créer un programme permettant de faire du calcul sur des fractions, en donnant les résultats sous forme de fraction réduite.

Le sujet du projet

Cahier des charges de base

- Proposer un type de données pour représenter une fraction.
- Écrire un programme qui étant donné deux fractions rentrées sous la forme ci-dessus, et un opérateur, renvoie le résultat.
- Écrire un programme utilisant la méthode de Héron pour calculer la valeur approchée d'une racine carrée sous forme de fraction, la précision étant donnée.
  http://fr.wikipedia.org/wiki/M%C3%A9thode_de_H%C3%A9ron
- Programmation d'une interface :
  - Rentrer au clavier une opération à faire, composée soit de deux fractions séparées par un opérateur (sans espaces), soit d'un formalisme à définir pour le calcul d'une valeur approchée de la racine carrée.
  - Affichage du résultat

Pour aller plus loin

- Compléter par un programme donnant l'écriture en fraction continue d'un nombre rationnel, ainsi que l'interface correspondante

7. Algorithme de Pledge **(*)

L'algorithme est simple à programmer. Par contre l'interface graphique donne plusieurs challenges

Présentation du problème.

L'algorithme de Pledge permet de sortir d'un labyrinthe, lorsque l'on part d'un point quelconque à l'intérieur. Un labyrinthe peut toujours être représenté sous forme de graphe, il est donc assez simple de programmer cet algorithme.

http://interstices.info/jcms/c_46065/l-algorithme-de-pledge
http://en.wikipedia.org/wiki/Maze_solving_algorithm#Pledge_algorithm

Le sujet du projet

Cahier des charges de base

- Programmer l'algorithme de Pledge étant donné un graphe représentant un labyrinthe, on considérera que tous les murs sont à angle droit.
- Interface :
    - Étant donné un graphe représentant un labyrinthe, donner le sommet représentant la position initiale ainsi que le sommet représentant la sortie
    - Afficher la suite des directions permettant de sortir du labyrinthe

8. Stéganographie et traitement des images **

Sujet d'ISN, car il y a un chapitre images en ISN qu'il n'y a pas en NSI. L'algorithme est assez simple à programmer, une fois qu'on l'a compris, et qu'on a compris comment les images sont codées sur ordinateur...

Présentation du problème.

La stéganographie est « l'art de la dissimulation ». Il s'agit de cacher une image dans une autre (ou un son dans un autre etc…). Soit pour jouer aux espions, soit pour mettre un copyright sur une image

Le sujet du projet

Cahier des charges de base

- proposer un algorithme qui, étant donné deux images dans un format simple non compressé, l'une étant « lourde » et l'autre « légère » (en taille), cache l'image légère dans la lourde. On utilisera la technique du bit de poids faible (cf http://fr.wikipedia.org/wiki/St%C3%A9ganographie). On réfléchira sur la taille relative des images.
- Il est impératif de travailler à partir des outils proposés par le professeur (programme « utilitaires images » et format ppm de préférence). Sinon cela devient vite compliqué. Par ailleurs, travailler dans un format compressé (comme le jpg) détruit l'image cachée.
- Proposer un autre algorithme qui extrait une image cachée dans une autre (ie l'opération inverse du précédent)

Pour aller plus loin

- compléter par d'autres effets sur une image (par exemple le contour avec des masques sophistiqués).

9. Cryptanalyse du chiffre de Vigénère *

Présentation du problème.

Le chiffre de Vigénère est une méthode pour rendre un message secret. Elle a été popularisée par Blaise de Vigenère, diplomate, cryptographe, alchimiste et astrologue français du 16ème siècle. Le véritable inventeur en est Giovan Battista Bellaso, cryptographe du 16ème siècle également, dont on ne sait presque rien. Cette méthode a été utilisée pendant plus de trois siècles. Dans ce projet, quasiment pas d'interface graphique.

Voir http ://fr.wikipedia.org/wiki/Chiffre_de_Vigen%C3%A8re

Le sujet du projet

Cahier des charges de base

- Ecrire un premier programme réalisant le chiffrage de Vigenère d'un texte en langue française écrit en majuscules, sans signe diacritique ni espace (c'est-à-dire n'utilisant que les lettres majuscules non accentuées de l'alphabet, et rien d'autre), en utilisant une clef donnée par l'utilisateur sous la même forme
- Ecrire un second programme réalisant le déchiffrement d'un texte crypté par le premier programme, en utilisant la clef ayant servi au chiffrement
- Ecrire un troisième programme effectuant la cryptanalyse d'un texte crypté par le premier programme, sans la connaissance de la clef de cryptage, en utilisant la méthode de l'indice de coïncidence décrite ici : http://fr.wikipedia.org/wiki/Indice_de_co%C3%AFncidence , ainsi que dans l'article sur le chiffre de Vigénère.
- Programmation d'une interface pour chacun des 3 programmes :
    - Saisir un texte et sa clef de chiffrement pour les 2 premiers programmes et sans clef pour le troisième
    - Afficher le texte résultant pour les 2 premiers programmes et un ensemble de textes possibles pour le troisième
    - Réaliser une interface unique pour les 3 programmes

Pour aller plus loin :

- Ecrire un troisième programme effectuant la cryptanalyse d'un texte crypté par le premier programme, sans la connaissance de la clef de cryptage, en utilisant la méthode de Kasiski et Babbage décrite ici : http://fr.wikipedia.org/wiki/Cryptanalyse_du_chiffre_de_Vigen%C3%A8re
- Ecrire un programme permettant transfromant un texte standard en texte sans signes diacritiques.

10. Snake **

Se prête bien à la programmation objet.

Présentation du problème.

Le but est de programmer le jeu classique du serpent.

Le sujet du projet

Cahier des charges de base

- Écrire le jeu du serpent avec une interface graphique, et les fruits de base, si nécessaire en programmation objet.

Pour aller plus loin

- Rajouter les fruits qui font changer de sens, ainsi que d'autres effets amusants (vitesse par exemple)

11. Casse-brique **

Se prête bien à la programmation objet.

Présentation du problème.

Le but est de programmer le jeu classique du casse-brique. Ce projet me paraît un peu plus difficile que d'autres.

Le sujet du projet

Cahier des charges de base

- Écrire le jeu du casse-brique avec une interface graphique, si nécessaire en programmation objet.

Pour aller plus loin

- Rajouter des effets sur certaines briques (taille de la raquette, accélération, ralentissement de la balle)

12. 2048 **(*)

Présentation du problème.

Le but est de programmer le jeu « 2048 ».

Le sujet du projet

Cahier des charges de base

- Écrire le jeu 2048… Pour ceux qui ne connaissent pas, tester le jeu sur téléphone portable avant. Les règles sont très simples

Pour aller plus loin

- Jouer avec des images plutôt que des nombres, sur interface graphique.

13. Tétris ***

Tout est dur dans ce projet. A commencer de préférence directement en programmation objet

14. Tower défense **

Du style minimaliste, avec un seul type de tours et une seule vague d'ennemis. A commencer de préférence directement en programmation objet. Peut être approfondi avec plusieurs vagues/types d'ennemis/tours. Interface graphique obligatoire bien sûr.

Autres projets non détaillés.

Les difficultés sont estimées, ces projets n'ayant pas été réalisés par d'anciens élèves d'ISN

Jeux de plateau

- un jeu du type "Mastermind". L'ordinateur fait deviner à l'humain. *(*)
- un jeu du type "démineur". L'ordinateur fait deviner à l'humain. **
- un jeu du type "Reversi/Othello" avec une interface graphique
- un arbitre de jeu d'échec : deux joueurs jouent et l'ordinateur renvoie si le coup est valable, s'il y a échec, et qui a gagné. *(*)

- un jeu du type " Motus". L'ordinateur fait deviner à l'humain. *
- un jeu du type "Abalone" (humain contre humain) *(*)
- un jeu du type "Backgammon" (humain contre humain) *(*)
- un jeu de type "Pipopipette" (humain contre humain / humain contre ordinateur ???) */**

Jeux d'arcade

- space invaders

Simulateurs

- De rubiks cube : le patron du rubik's cube est tracé et l'utilisateur peut simuler les mouvements du rubik's cube
- programme permettant de reproduire la vision des daltoniens

Cryptage de données, représentation de l'information

Les deux projets suivants sont en rapport avec le programme de mathématiques approfondies de terminale. Ils sont très faciles du point de vue informatique, pas beaucoup plus durs du point de vue mathématique.

- Cryptage RSA ½ *
- Machine Engima (* ?)
- Générateur de QR-code, de code barre *(*)

Autres

- Images : transformation du photomaton : https://fr.wikipedia.org/wiki/Transformation_du_clich%C3%A9_Photomaton *
- Générateur de labyrinthe. Peut être combiné avec l'algorithme de Pledge. *