PIX-info: Jouons (3): Motus

Grenoble INP – Esisar – Academic year 2025-26 • Compiled on 30 novembre 2025

Version du 30 novembre 2025

Objectif pédagogique

- Pratiquer l'algorithmique et la programmation C
- Gagner de l'autonomie dans la conception et les tests logiciels
- Pratiquer le vocabulaire de lecture de programmes
- Pratiquer la fiche de syntaxe C "niveau 1".
- **Premier usage de la compilations séparée et de Makefile**
- Proposer une **amélioration d'algorithme** (analyse de complexité).

Ni le code ni les réponses ne sont évaluées, l'objectif est de **pratiquer**.

Description :

On désire réaliser un jeu de type motus: cf (https://motus.absolu-puzzle.com/ dans lequel:

- l'ordinateur tire un mot de taille 6 au hasard dans un dictionnaire, et fournit la première lettre.
- la joueuse doit trouver ce mot, en proposant des mots:
  - de taille 6
  - présents dans le dictionnaire
- lors d'une tentative, si le mot proposé est valide, l'ordinateur indique pour chaque caractère s'il est bien placé, mal placé ou non utilisé dans le mot à deviner.
- la joueuse dispose d'un nombre de tentatives limité.

Il vous est fourni un code à trous, sous la forme d'une archive comportant les fichiers suivants:

`dico-fr.txt  main.c  Makefile  motus.c  motus.h`

Pour compiler les fichiers, vous utiliserez la commande `make` dans le terminal, qui réalise automatiquement la compilation séparée des fichiers `motus.c` et `main.c`. Les deux concepts de programmation utilisés sont la compilation séparée (cf , et les slides dispo sur chamilo (2A,CS221)), ainsi que l'automatisation de tâches dépendantes par l'utilisateure `make` (`man make`, ainsi que Wikipedia). Il n'est pas demandé ici de comprendre en détail comment fonctionnent ces concepts, à part les quelques questions proposées plus bas.

Questions préliminaires de compréhension du code, et d'analyse

Répondre aux questions suivantes:

- A quoi sert le fichier `motus.h`?
- Si on modifie `motus.c`, quelles fichiers doivent-être recompilés (tester avec `make`).
- Quelles variables sont globales dans ce programme ? Dans quel(s) fichier(s) sont-elles déclarées ? Utilisées ?
- Quelle fonction permet d'initialiser le dictionnaire à partir du fichier ? Expliquez comment seuls les mots de taille 6 (MAXLETTRE) sont stockés dans le dictionnaire. Expliquez pourquoi les lignes du dictionaire sont plus grandes que MAXLETTRE.
- Rappeler la complexité de la recherche d'un mot dans le dictionnaire, avec un parcours séquentiel simple.
- Rappeler la complexité de la recherche d'un mot dans le dictionnaire, avec un parcours utilisant le fait que le dictionnaire est trié. Ces deux questions doivent être rédigées sur papier.

Travail à faire

Pour l'implémentation, il est recommandé d'utiliser au maximum les fonctions de chaînes.

- Bien comprendre le code fourni, et commencer par `chargeMots`, vérifier que vous obtenez bien une représentation mémoire du contenu du fichier.
- Implémenter la recherche simple (linéaire) dans `chercheMot` et tester.
- Implémenter l'impression du tableau statut, tester.
- Réaliser la boucle de jeu, et le jeu.

Si vous avez du temps

- Implémenter la recherche dichotomique du mot dans le dictionnaire.

Pour aller plus loin, si vous avez terminé les 3 jeux (difficile)

- Choisir un des trois jeux, et réaliser une interface graphique avec la bibliothèque `libsx`:
  - installer le paquet `libsx-dev`, ou alors récupérer les fichiers sur le web et les mettre dans le répertoire de votre jeu
  - utiliser `-lsx` dans la ligne de compilation
  - éventuellement se reporter à cette excellente explication
  - n'oubliez pas de faire un dessin de votre interface graphique avant.