Interface de programmation/ API

|  |  |  |
| --- | --- | --- |
| interface | Paramètre de sélection | commentaire |
| Menu ( ) | Menu de sélection   1. Lecture du message 2. Stockage du message 3. Cryptage/Décryptage 4. Quitter | * Pour la partie « menu », la fonction utilisé ici est la fonction « menu( ) : », c’est dans cette partie que nous ferons interagir toutes autres fonctions du programme. * La fonction « encrypt(letter, key, selector\_direction) : » nous permettra de réaliser l’encryptage du message. |
| Logique | Paramètre/ Sélection | Commentaires |
| EnigmaMachine | 1. Rotor = ‘I II III’ 2. Reflector = ‘B’ 3. MessageIn 4. MessageOut | * La fonction ” rotor\_permute(rotor\_name, key):” est celle qui nous permettra de faire tourner les rotor le choix des rotors étant dixé par défaut * En ce qui concerne les reflector, la fonction « init\_rotor\_and\_reflector(rotors, Reflecteurs) : » initialisera les rotor et les reflecteurs |
| Data | Paramètre/ Selection | Commentaires |
| Sauvegarde message | Fichier de sauvegarde  Message à sauvegarder | * Pour ce qui de l’enregistrement du message, c’est la fonction « save\_message(file\_name, message\_to\_send) : » qui se chargera d’enregistrer le message qui sera envoyé |
| Lecture Message | Fichier de lecture  Message à lire | * En ce qui concerne la lecture du message une fonction « lecture\_message( filename, result) » a été mise en œuvre pour nous diriger vers le dossier où se situe le message |
| Initialise Rotors | 1. Rotor = ‘I II III’ 2. Reflector = ‘B’ | L’initialisation des rotors lisent le fichier « json.load(p) » |

Dans cette rubrique nous présenterons l’organisation en couche des parties Application, logique et data sous forme de tableau. Il sera mis ici en exergue la fonction que chaque partie devra réaliser.