Rapport Copilot :

En écrivant la classe Test : le main a été proposé par Copilot avec un print hello world dedans, cela a été automatique.
En écrivant des commentaires, Copilot propose de compléter avec ses mots dont un squelette pour spécifier la méthode, il arrive à proposer de fermer les commentaires.
Copilot ne propose pas de correction une fois la méthode écrit même après changement des commentaires.
On peut voir quelques différences entre ce qui était attendu dans le cours et ce qu'on obtient maintenant de copilot :
Il était attendue : /* Une étiquette avec pour texte "Hello World !" */
J'ai eu : /* Un bouton avec pour texte "Hello World !" */

Copilot ajoute l'extension de la bibliothèque pour le bouton et le Jframe lui même. Il est même capable d'utiliser les API données en commentaires comme java.util.random. Il apprend à écrire selon les classes déjà définies auparavant, constructeur avec argument point dans Parcours.
Même Avec peu d'information il est capable de définir de bonnes constantes X_MIN et X_Max, il fallait juste modifier la valeur mais il n'avait pas de contexte.
Il y a eu quelques ratés dont l'écriture de l'algorithme de génération de points, j'ai eu une boucle infini à cause de random qui donnait toujours la même valeur et qui n'augmentait pas.
J'ai écrit du code nécessitant un accesseur pas encore écrit, Copilot le détecte et le propose de suite avec le commentaire adéquat.
Copilot devient biaisé avec ce que j'écris de plus récent, il répète mes erreurs.
Copilot complète plutôt bien les constantes sur les fenêtres, je venais de définir ratio il l'a tout de suite proposé.
Création d'une nouvelle classe, après avoir extends thread écrit direct le constructeur nécessite un accesseur qui modifie avancement similaire à Descendre. Il le propose tout seul quand je vais dans position et il propose même le commentaire.
Pour la nouvelle fonctionnalité, il a fait super bien fait le score et le menu ça a pris 5/10 mins pour un truc basique qui fonctionne, j'ai suivi le même principe que pour l'écriture de Test.
Pour écrire score, il a juste fallu donner un commentaire au début pour décrire le résultat voulu et ajouter l'extends Thread pour que Copilot complète lui même le reste.
Pour écrire le constructeur, il propose un this.start() pour lancer le thread au moment de sa création ce que je ne veux pas mais c'est intéressant, Copilot peut s'adapter en fonction de la classe étendue.
Je n'ai pas suivi de méthode particulière pour utiliser Copilot, je mettais des commentaires pour tester ce qu'il est capable de faire entre les accesseurs simples et des algorithmes dotés de boucles. Il y a eu plus de difficultés pour écrire des algorithmes corrects mais dans l'ensemble c'est très efficace.