

AC 13.02 – Lire, exécuter, corriger et modifier un programme.

Ressources: SAE 1.05 - R1.07

Contexte : Dans le cadre de ma formation en BUT Réseaux et Télécommunications, j'ai réalisé une activité liée à la compétence AC13.02 – Lire, exécuter, corriger et modifier un programme.

L'objectif était de lire de petits programmes, de les exécuter et de corriger des erreurs simples. L'objectif était également de reprendre ou compléter des scripts fournis pour lire, transformer ou présenter des données (dans un tableau ou un fichier).

Savoir mis en œuvre → Connaître les bases du Python et comprendre le déroulement d'un programme.

Savoir-faire mis en œuvre → Lire et comprendre un petit programme fourni. Comprendre et corriger les erreurs potentielles des programmes que j'ai réalisés en autonomie. Savoir exécuter le programme dans un environnement adapté et vérifier le bon fonctionnement de ce dernier (avec des tests). Compléter un programme partiellement donné et réussir à le faire fonctionner.

Savoir-être mis en œuvre → Faire preuve de rigueur dans la lecture et la modification du programme (éviter une perte de temps voire de ne pas voir le problème).

Tâche réalisée et les résultats → J'ai dû reprendre des bouts de code voire des codes entiers pour les tester, les corriger si besoin, puis, dans certains cas, les incorporer dans un code plus important. En voici un exemple.

- Dans la ressource R1.07, j'ai dû, lors de mon projet final, récupérer des bouts de code, les lire et les comprendre, les tester pour voir s'ils fonctionnaient bien puis les compléter avec d'autres codes.

Dans le code suivant (en langage C), les fonctions deux des fonctions étaient données et les fonctions "tri_sélection" et "tri_bulles" étaient à compléter. Il fallait alors comprendre les fonctions données pour créer celles manquantes, ainsi que tester chaque fonction pour veiller à son bon fonctionnement.

```

#include <stdlib.h>
#include <time.h>
#include <stdbool.h>
#include <stdio.h>

void echanger(int* a, int* b){
    int c = *b;
    *b = *a;
    *a = c;
}

bool valid_tri(int T[], int n){
    for (int i = 0; i < n - 1; i++){
        if (T[i] > T[i + 1]){
            return false;
        }
    }
    return true;
}

void afficher_tab(int T[], int n){
    printf("Les tableau trié est ");
    for (int i = 0; i < n; i++){
        printf("%d\n", T[i]);
    }
}

void remplir_tab(int T[], int n){
    srand(time(NULL));
    for (int i = 0; i < n; i++){
        T[i] = rand() % 100; // nombre entre 0 et 99
    }
}

void copie_tab(int dest[], int T[], int n){
    for (int i = 0; i < n; i++){
        dest[i] = T[i];
    }
}

void tri_selection(int T[], int n){
    for (int i = 0; i < n - 1; i++){
        int indice_min = i;
        for (int j = i + 1; j < n; j++){
            if (T[j] < T[indice_min]){
                indice_min = j;
            }
        }
        echanger(T+i, T+indice_min);
    }
}

void tri_bulles(int T[], int n){
    bool ech = true;
    while (n > 0 && ech==true){
        ech = false;
        int j = 0;
        while (j < n-1){
            if (T[j] > T[j + 1]){
                echanger(&T[j], &T[j + 1]);
                ech = true;
            }
            j++;
        }
        n--;
    }
}

```

- Comme dit précédemment, j'ai aussi dû corriger les erreurs éventuelles des programmes que je réalisais moi-même (problème de logique, d'indentation ou de compréhension de la consigne).

Les problèmes que j'ai rencontrés → J'ai pu rencontrer des problèmes de compréhension des consignes pour des programmes parfois compliqués sur des sujets que je ne connaissais pas. Mis à part cela, pas de problème en particulier.

Point faible / Point à améliorer → Je pense pouvoir améliorer ma capacité à résoudre les erreurs dans un code par moi-même, sans devoir aller chercher sur internet ou demander à une IA.