Examen : C et UNIX

Dr. Jehan-Antoine Vayssade

20 mars 2025

Instructions

— **Durée :** 2 heures
— **Note totale / maximale :** 20 points
— **Des points bonus peuvent être cachés !**
— **Les réponses doivent être argumentées**
— **Prenez uniquement un stylo**
— **Attention aux pièges**

Compréhension du langage C

Question (1 point)

Quelle sera la sortie du code suivant ? Expliquez pourquoi, par étapes.

```c
int x = 0;
int y = 3;

if (y == x)
    printf("Condition␣vérifiée\n");

if (y != x)
    printf("Erreur␣inattendue\n");
```

Question (1 point)

Comment s'appellent les zones mémoires associées à chaque variable ?

```c
const char *message = "La␣choucroute␣avec␣la␣biere";

int main() {
    int status = 5;
    int *result = (int*) malloc(sizeof(int));
}
```

Question (1 point)

Quelle est la taille de la structure suivante ? Expliquez pourquoi. Existe-t-il un type de base qui pourrait stocker la totalité des informations de cette structure ? Si oui, lequel et quel serait l'intérêt ? Sinon, pourquoi ?

```c
struct Data {
    char x;
    short y;
};
```

Question (2 points)

Écrivez une fonction en C qui parcourt une chaîne de caractères avec un pointeur et compte le nombre de voyelles (a, e, i, o, u, en minuscules uniquement). Exemple : "hello" retourne 2. La chaîne se termine par '\0'. La fonction devra avoir la signature suivante :

```c
int count_vowels(char *str) {
    int count = 0;
    // TODO
    return count;
}
```

Question (1 point)

Quelle erreur est faite dans le code suivant ? Vous pouvez vous aider d'un dessin. Que va-t-il se passer à l'exécution ? Qu'est ce que le programmeur a écris dans x ?

```
1        int main()
2        {
3            int *x = 0;
4            x = (int*)&x;
5            *x = 5;
6            return 0;
7        }
```

Question (2 points)

Èrcivez une fonction en C qui retourne la valeur absolue d'un entier de façon optimisée.

Question (1 point)

Expliquez la fonction suivante, que se passe-t-il lors de son utilisation ? Étape par étape.

```
1        inline int swap(int a, int b) {
2            int tmp = a;
3            a = b;
4            b = tmp;
5        }
6
7        int x = 0, y = 2;
8        swap(x, y); // Que vaut x et y après ?
```

Question (1 point)

Expliquez le code suivant, quel est le risque ? Que vaut l'entier x après l'éxécution ?

```
1        typedef struct { char a, b, c, d; } octets;
2        int x = 0;
3        ((octet*)&x)->a = 0xAA;
4        ((octet*)&x)->b = 0xBB;
5        ((octet*)&x)->c = 0xCC;
6        ((octet*)&x)->d = 0xDD;
```

Synchronisation des processus

Documentation

```c
#include <pthread.h>
#include <semaphore.h>

int pthread_create(pthread_t *thread, const pthread_attr_t *attr,
   void *(*start_routine)(void*), void *arg);
int pthread_cancel(pthread_t thread);
int pthread_join(pthread_t thread, void **value_ptr);
void pthread_exit(void *value_ptr);

pthread_mutex_t mutex = PTHREAD_MUTEX_INITIALIZER;
int pthread_mutex_destroy(pthread_mutex_t *mutex);
int pthread_mutex_lock(pthread_mutex_t *mutex);
int pthread_mutex_trylock(pthread_mutex_t *mutex);
int pthread_mutex_unlock(pthread_mutex_t *mutex);

int sem_init(sem_t *sem, int pshared, unsigned int value);
int sem_destroy(sem_t *sem);
int sem_wait(sem_t *sem);
int sem_post(sem_t *sem);
```

Question (2 points)

Expliquez ce qu'est une section critique. Apportez un exemple, proposez une analogie du monde réel. Que se passe-t-il quand deux threads modifient une même variable ? (ex : ++ et −−).

Question (5 points)

Un programme d'analyse d'image embarqué dans un robot agricole (WeedElec) rencontre des problèmes de performances. Son but est de détecter, classer et éliminer des adventices (mauvaises herbes). Actuellement, chaque étape du programme est exécutée séquentiellement. Notamment : (1) capture de l'image, (2) segmentation sol/végétation, (3) détection des plantes, (4) classification et (5) activation d'un laser qui élimine la plante non désirée. Fort heureusement, ce code est exécuté sur une puce "Arm Cortex-A78AE" (6 curs) et un GPU pour l'utilisation du deep learning. Idéalement, seule l'image la plus récente est traitée. `Si vous pensez ne pas avoir le temps, proposez un schéma de synchronisation, le plus détaillé possible (2 pts).`

Question (3 points)

Après une mise à jour du robot, d'autres lasers ont été installés pour pallier les problèmes de vitesse d'extermination et de maintenance. En effet, le laser n'a pas le temps de refroidir, impliquant une surchauffe constante qui le fait griller fréquemment. On vous demande donc de paralléliser la fonction "kill_weed", chaque thread éliminant une plante toutes les secondes. `Si vous pensez ne pas avoir le temps, proposez une idée vue en TP (1 pt).`

Code de la question à 5 points

```c
typedef struct {
    float* raw;
    int width, height, channel;
} MultiSpectralImageData;

typedef struct {
    unsigned int x, y, w, h; // rectangle de détection
} vec4i;

typedef struct {
    MultiSpectralImageData *image;
    unsigned char *segmented;
    vec4i *plant_positions;
    int *classifications; // 0 = adventice ; 1 = gentile plante (miam)
    int plant_count;
} ProcessedData;

int main() {
    while(1) {
        ProcessedData *data = new_processing_data();
        // Allocation, Capture et copy des données multi-spectral
        camera_capture(data);
        // Allocation de @segmented, Calcule basé un NDVI
        segment_image(data);
        // Allocation de @plant_positions, détecte les plantes
        detect_plants(data);
        // Classification des plantes par un réseau de neurones
        classify_plants(data);
        // Extermination des mauvaise herbes
        kill_weed(data);
        // Libération des ressources pour la prochaine itération
        destroy_processing_data(data);
    }

    return 0;
}
```

Code de la question à 3 points

```c
#define LASER_COUNT 10

void kill_weed(ProcessedData *data) {
    for (int i = 0; i < data.plant_count; i++) {
        if (data.classifications[i] == 0) {
            // Activation du laser 0 à la position de la plante i
            laser_target(0, proc_data.plant_positions[i]);
        }
    }
}
```