Algorithmes et Pensée Computationnelle

Logiciels système

Le but de cette séance est de comprendre le rôle d'un système d'exploitation, de logiciels système, d'interpréteurs, de compilateurs et des librairies. Les exercices présentés au cours de cette séance se présentent sous deux sections. Une section composée de questions à choix multiples et une section permettant de découvrir et se familiariser avec votre futur environnement de travail.

1 Systèmes d'exploitation

Le système d'exploitation (OS) est le logiciel le plus important de l'ordinateur car il est responsable de tous les autres programmes et de leurs priorités d'exécution, il organise aussi les interactions et l'orchestration des différents composants physiques de la machine.

L'OS contrôle l'accès au matériel physique et gère les processeurs, la mémoire, le stockage, la sécurité et les périphériques externes. Par exemple, l'OS identifie quel utilisateur est connecté (au moment d'entrer le mot de passe à l'allumage), il reconnaît quelles touches du clavier et de la souris sont pressées, il affiche les images à l'écran et il sauvegarde les fichiers dans les disques durs internes ou externes.

La plupart du temps, de nombreuses applications sont exécutées au même moment sur le même ordinateur. Elles ont besoin d'utiliser les mêmes composants de l'ordinateur (CPU, RAM, Storage). Cependant, ces différents composants ne peuvent faire qu'une seule chose à la fois (même si ils les font très rapidement), c'est la raison pour laquelle l'OS orchestre chaque tâche de manière à ce que l'utilisateur ait l'impression que tout se déroule en même temps.

Question 1: (🕓 *5 minutes*) **QCM**
Lequel de ces logiciels n'est pas un système d'exploitation ?

1. Microsoft Office 2012
2. Microsoft Windows 98
3. Unix BSD
4. Gentoo
5. Aucune réponse correcte.

Question 2: (🕓 *5 minutes*) Quel est le rôle du noyau du système d'exploitation *Kernel* ? Trouvez la **mauvaise** réponse.

1. Servir d'interface de communication entre la partie logicielle et matérielle
2. Gérer les ressources physiques de l'ordinateur
3. Fournir les mécanismes d'abstraction du matériel
4. Interpréter les instructions et les convertir en binaire
5. Toutes les réponses sont correctes.

Question 3: (🕓 *5 minutes*) Que représente la valeur **CPU Time** donnée par le gestionnaire de tâches de votre système d'exploitation ?

1. Temps d'utilisation active de l'ensemble des cores du processeur.
2. Temps réel d'utilisation d'une partie des ressources du processeur.
3. Temps d'utilisation de l'ordinateur depuis le dernier démarrage.
4. Temps maximum dédié à un processus particulier.
5. Nombre de secondes écoulées depuis l'EPOCH.

Informations utiles

— L'EPOCH représente la date initiale à partir de laquelle est mesuré le temps par les systèmes d'exploitation.
— Pour accéder au gestionnaire de tâches sous Windows, ouvrez le menu démarrer ¿ Saisissez "Gestionnaire de tâches". Sous Mac, faites "command+espace" et saisissez "Moniteur d'activité'.

Solution

Temps d'utilisation active de l'ensemble des cores du processeur. Exemple : Imaginez qu'un processus actif utilise constamment 10% de la puisesance du processeur pendant 20 minutes. Le temps compté et affiché dans le gestionnaire de tâches sera de 2 minutes.

Question 4: (*5 minutes*) Que fait la commande suivante : **ls** *(Linux/MacOS)* / **dir** *(Windows)* ?

1. Liste l'ensemble des fichiers du disque.
2. Affiche sous forme de liste l'ensemble des processus en cours.
3. Retourne uniquement les dossiers du répertoire courant.
4. Affiche tous les dossiers et fichiers du répertoire courant.
5. Aucune réponse n'est correcte.

Solution

ls *(Linux/MacOS)* ou **dir** *(Windows)* permet d'afficher le contenu du répertoire courant.

Conseil

Pour avoir une description détaillée d'une commande, vous pouvez ajouter **man** devant chaque commande sous Linux/MacOS ou ajouter **–h**, **––help** ou **/?** après chaque commande sous Windows.

2 Prise en main de l'environnement de travail

2.1 Installation des outils *(30 minutes)*

Suivre le guide suivant pour installer les outils qui seront utilisés lors des prochaines séances de TP. //-TODO : Lien vers le document prerequisite.

3 Interpréteurs et compilateurs

Les ordinateurs ne "comprennent" pas les langages de programmation, il faut passer par un programme qui va convertir le code écrit par un humain en instructions que l'ordinateur comprend (à base de 0 et de 1).
Les interprètes et les compilateurs servent à faire ce travail de traduction, ils transforment donc les langages de programmation qui sont faits pour être compris et écrits par des humains, vers des instructions compréhensibles pour des ordinateurs.
La façon dont les interprètes et les compilateurs opèrent est différente et les deux approches apportent leur propre lot de bénéfices et d'inconvénients :

Compilateurs	Interpréteurs
Les COMPILATEURS traduisent à l'avance	Les INTERPRÈTES traduisent au fur et à mesure
Programmes générés à l'avance (besoin d'anticiper tous les cas de figure)	Programmes générés au fur et à mesure
Le programme est intégralement traduit à l'avance	Les instructions sont traduites à la volée une par une
Le compilateur trouve les erreurs à la compilation	L'interprète relève les erreurs pendant l'exécution du programme
Plus rapide	Plus lent
Code machine généré et optimisé à l'avance	Code machine généré à la volée et optimisé à chaque exécution
Ex : C, C++, Java, Scala, Rust, etc.	Ex : Bash, Python, Javascript, etc.

Question 5: (*10 minutes*) En utilisant l'invite de commande (Terminal), créer un fichier contenant l'instruction suivante :

```
1  if __name__ == "__main__":
2      print("Hello world")
```

Enregistrer le fichier sous **hello.py**. Utiliser l'interpréteur de Python pour exécuter le programme que vous venez de créer.

Conseil

- Assurez-vous d'avoir correctement installer les outils de développement avant d'exécuter le programme.
- Utiliser un éditeur de texte présent sur votre ordinateur pour écrire directement votre programme depuis le terminal sous MacOS ou Linux. Nous vous conseillons d'utiliser Nano en tapant **nano hello.py**.
- Sous Windows, vous pouvez utiliser n'importe quel éditeur de texte. "Notepad" est installé par défaut sous Windows.

4 Librairies

Une librairie est un ensemble de fonctions qui ont pour but d'être utilisées par d'autres programmes, mais une librairie seule ne suffit pas pour faire un programme. Par exemple, que ce soit Google Chrome, Word, ou Instagram, presque tous les programmes ont besoin d'utiliser des listes d'objets. Au lieu de réécrire l'ensemble des fonctions qui permettent de créer des listes et d'interragir avec, ces différents programmes utilisent tous une librairie écrite par un tiers.

En Python, de nombreuses librairies sont disponibles de base (list, set, dict, str, etc.) et de nombreuses autres peuvent être utilisées en utilisant le mot-clé **import**.

Dans l'exemple suivant, nous importons la librairie math qui propose de nombreuses fonctions mathématiques et nous essayons la fonction factorielle pour calculer **5!**.

Exemple

```
1  import math
2
3  fact = math.factorial(5)
4  print(fact)
```