Année-scolaire : 2020-2021

RAPPORT DU TP DE SYSTEME D’EXPLOITATION

Participants

Bhale Silvaire

Konan Abraham Kouassi Obed

Afraitan Ayal

Professeur : Prof. Soro

Sommaire du questionnaire

[ Question N°1 : Ecriture de scripts \* 3](#_Toc77851014)

[1-Ecrire un script qui calcule le PGCD de deux entiers non nuls entrés au clavier 3](#_Toc77851015)

[Écrire une commande listebranche en langage C qui affiche le contenu de tous les répertoires qui figurent dans le chemin de la référence absolue de votre répertoire courant 3](#_Toc77851016)

[ Question N°2 : Archivage de fichiers \* 4](#_Toc77851017)

[1-Créer à l’aide gedit quatre fichiers : fic1 contenant le mot "Ceci", fic2 contenant le mot "est", fic3 contenant le mot "une", et fic4 contenant le mot "archive".. 4](#_Toc77851018)

[2- Archiver ces quatre fichiers dans une archive que vous nommerez essai.tar. 4](#_Toc77851019)

[ Question N°3 : Création de répertoires \* 4](#_Toc77851020)

[ Question N°4 : Consultation des attributs d’un fichier \* 5](#_Toc77851021)

[ Question N°5 : Création de processus \* 5](#_Toc77851026)

[3- Etudier la fonction fork()………………………………………………………………………………………………..5](#_Toc77851027)

[ Question N°6 : Communication entre processus par tube sans nom \* 5](#_Toc77851032)

[ Question N° 7 : Communication bidirectionnelle entre deux tubes\* 6](#_Toc77851033)

* Question N°1 : Ecriture de scripts \*

1. Ecrire un script qui calcule le PGCD de deux entiers non nuls entrés au clavier

***Analyse des données***

*Pour faire le calcul du PGCD de deux nombres entiers*

*Nous avons en entré deux entiers naturel non tous nuls notés a, b et en sortie un entier.*

***Action à faire***

*Nous appliquons les formules mathématiques*

*Nous gérons les cas particuliers suivant : si l’un des deux nombres est nuls ce cas se résout simplement en effet si l’un des nombres on déclare le second comme étant le PGCD, et si les deux nombres sont égaux alors on retourne l’un d’entre eux comme le PGCD.*

*Si a>b avec b non nul (I°) :*

*PGCD (a, b) =PGCD (b, r) où r est le reste de la division euclidienne d’a par b nous réitérons la même procédure jusqu’à ce que le reste r soit nulle la valeur b restant est le PGCD des deux nombres rechercher*

*Si a<b : on aura qu’a permuter les deux variables et revenir en (I°)*

*Conception du programme*

*Nous utilisons le langage de programmation en C*

*Nous déclarons les variables a, b, r entières pour les différentes variables définies dans les formules mathématiques et afin de gérer le cas où a<b on déclare une variable tmp ; pour cette permutation.*

1. Écrire une commande listebranche en langage C qui affiche le contenu de tous les répertoires qui figurent dans le chemin de la référence absolue de votre répertoire courant

***Analyse des données***

Pour afficher de façon récursive le contenu d’un répertoire

Nous avons besoin en entré d’une chaine de caractère qui contiendras le nom du répertoire dont nous voulons faire le parcours, l’utilisateur. Comme nous sommes dans un environnement linux le nom du répertoire devra être sous la

forme : /home/nom\_utilisateur/….

Au risque de recevoir une erreur de segmentation et de voir le programme s’arrêter. Nous utiliserons les différentes bibliothèques suivantes : <stdio.h> ,<stdlib.h>,<unistd.h>,<string.h>,<sys/types.h>,<sys /stat.h>,<dirent.h>.

***Action à faire***

Nous utilisons les bibliothèques <dirent.h>,<unistd.h>,<sys /types.h>

Pour créer une variable fichier\_ouvert pour ouvrir le répertoire dont le nom est passer en argument a notre procédure afficher\_ contenu, si l’ouverture du répertoire est un succès nous lisons les entres successives du répertoire avec la fonction readdir() ,ensuite si cette entre existe nous remplissons un buffer de types stat défini dans le fichier d’entête <sys /stat.h>ce buffer est rempli grâce à la fonction stat() dont la primitive est définie dans <unistd.h>. Nous consultons le contenu du champ st\_mode du buffer grâce à la fonction IS\_DIR qui prend en entre le champ st\_mode d’un buffer stat et retourne 1 si celui-ci est un répertoire et 0 sinon ;

Au cas où l’entrée est un répertoire on utilise la fonction strcat() pour concaténer le nom du répertoire passer en paramètre avec le « / » et le nom de l’entrée qui est un répertoire on appelle enfin de façon récursive la procédure afficher\_contenu.

Au cas où l’entrée est un fichier non répertoire on affiche son nom

Nous appelons la procédure dans la fonction main().

* Question N°2 : Archivage de fichiers \*

1. Créer à l’aide gedit quatre fichiers : fic1 contenant le mot "Ceci", fic2 contenant le mot "est", fic3 contenant le mot "une", et fic4 contenant le mot "archive"..
2. Archiver ces quatre fichiers dans une archive que vous nommerez essai.tar.

*Analyse des données*

*Nous répondons en même temps au deux questions posées ci-dessus.*

Pour la création des fichiers nous passons 4 fichier

Nous tapons la commande gedit dans le shell en passant le nom des différents fichiers un à un en argument. Pour l’archivage des fichiers on dans un programme C la fonction system

* Question N°3 : Création de répertoires \*

1. Etudier l’appel-système de création d’un nouveau répertoire : mkdir ()
2. Ecrire un script qui automatise la création de 3 répertoires dont les noms sont donnés en argument répertoire.

Cet appel système permet de créer un répertoire dont le nom absolue est passé en paramètre avec le mode d’utilisation. Le répertoire doit être au moins un sous répertoire de /home si l’utilisateur désire créer un répertoire dans le répertoire au-dessus /home, il devra avoir les privilège requis pour l’exécution de la commande.

*Analyse des données*

En entre nous avons un nom de répertoire absolue sous forme de chaine de caractères et en retour de la fonction on a un entier qui est soit nul pour un succès et -1 pour un échec.

ACTION A FAIRE : pour la conception de ce programme nous importons les modules <fcntl.h> ,<sys/types.h>.

* Question N°4 : Consultation des attributs d’un fichier \*

*L’objectif de cet exercice est d’afficher les attributs d’un fichier*

Analyse des données

*Le script prend en argument un nom absolu de fichier et affiche sur le périphérie standard les différents attributs de fichier c’est-à-dire le type de fichier les différentes permissions selon les utilisateurs*

**Action à faire**

Importer le fichier d’entête <unistd.h> dans lequel est définie la fonction stat () qui prend comme paramètre le nom d’un fichier et un buffer de struct stat dans lequel va se trouver les différentes caractéristiques du fichier dont le nom est passé en argument ensuite on importe <sys /stat.h> et enfin

<sys/types.h>, nous nous servons des différents macros (S\_ISDIR ,S\_ISCHR , …) et les constantes (S\_IRUSR ,S\_IWUSR,…) Pour trouver le type et les permissions sur le fichier passer en paramètre.

Le résultat de l’exécution du script avec le passage du fichier l3essai1 et avec /etc/passwd est sur l’image ci-dessous

* Question N°5 : Création de processus \*

1. Etudier la fonction fork ().

Objectif utilisation de la fonction fork () pour la création de processus

Analyse de données

*Pas de données en entrées*

*Cette fonction permet la création dynamique d'un nouveau processus qui s'exécute de façon concurrente avec le processus qui l'a créé, elle retourne 0 en cas de succès et -1 en cas d’échec*

*ACTION A FAIRE : importer le module <sys/types.h> dans lequel se trouve la définition du type de retour de la fonction fork () qui est le PID du processus crée on appelle aussi la fonction get\_pid() qui nous permettra de récupérer le PID du processus parent.*

* Question N°6 : Communication entre processus par tube sans nom \*

L’objectif est de faire communiquer deux processus à travers un tube sans nom

*Analyse des données*

Nous avons en entré le nom absolue d’un fichier texte contenant du texte et en sorti nous avons des parties du texte afficher sur la périphérie standard de sortie.

**Action à faire**

* Question N° 7 : Communication bidirectionnelle entre deux tubes\*

L’objectif est de faire communiquer deux processus à travers un tube sans nom

*Analyse des données*

Nous avons en entré le nom absolue d’un fichier texte contenant du texte et en sorti nous avons des parties du texte afficher sur la périphérie standard de sortie.

**Action à faire**

.