Tecnológico de Monterrey

Departamento **de Tecnologías de la Información**
Actividad 2**: 17 de Agosto de 2018**
Sistemas Operativos
Clave: TC2008 Grupo: 1
Campus Santa Fe

Profesor: Karen Azurim García Gamboa Calificación: _____

Nombre: _____ Matrícula: _____

Actividad 2: Llamadas al Sistema

Las llamadas al sistema proporcionan la interfaz entre el sistema operativo y un programa en ejecución. Linux permite ejecutar llamadas al sistema desde un programa en un lenguaje de alto nivel, en particular en C, en cuyo caso las llamadas se asemejan a llamadas a funciones definidas en una biblioteca estándar. Este laboratorio está enfocado en el uso de las funciones `exec()` y `system()` las cuales permiten realizar las llamadas al sistema `exec()` y `system()`, respectivamente.

Objetivo
El alumno se familiarizará con algunas de las llamadas al sistema como `exec()` y `system()` y entenderá su funcionamiento.

Desarrollo

Ejemplo 1. Uso del procedimiento `execl()` que ejecuta la llamada al sistema `exec()`.

```c
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>

int main(){
        int salida;
        printf("Ejemplo de exec \n");
        printf("aqui %s\n", getenv("USER"));
        salida = execl("/bin/ps", "/bin/ps", "-fu", getenv("USER"), NULL);
        printf("Salida del comando: %d\n", salida);
        exit(0);
}
```

Códifica el programa que muestra el uso de `exec()` y analiza su funcionamiento.

Ejemplo 2. Uso del procedimiento `system()`.

```c
#include <stdio.h>
#include <stdlib.h>

int main(){
        int salida;
        char command[100];
        printf("Ejemplo de la llamada system \n");
        sprintf(command, "/bin/ps -fu %s", getenv("USER"));
        salida = system(command);
        printf("Salida del comando: %d\n", salida);
        exit(0);
}
```

Códifica el programa que muestra el uso de `execl()` y analiza su funcionamiento.

Ejemplo 3. Uso de `fork()` y `exec()`.

```c
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>

int main(){
        printf("PID of mi programa fork.c = %d\n", getpid());
        pid_t p;
        p = fork();
        if (p==-1)
                printf("Error al crear un proceso hijo");
        if (p==0){
                printf("Estamos en el proceso hijo y ejecutamos hello.c \n");
                execl("./hello", "./hello", (char *)0);
        }else{
                printf("Estamos en el proceso padre");
        }
        return 0;
}
```

Códifica el programa que muestra el uso de `fork()` y `exec()`, analiza su funcionamiento. Deberás responder las preguntas después del análisis.

Responder

1. Con base en el análisis realizado de los ejemplos 1 y 2, explíca brevemente cuál es la diferencia entre `exec()` y `system()` y su funcionamiento.

2. Explica brevemente qué está haciendo el ejemplo 3.

3. ¿Podría la siguiente llamada regresar algún valor en **`cuenta`** distinto de **`nbytes`**? Si es así, explica ¿por qué?

   ```
   cuenta = write(fd, bufer, nbytes);
   ```

Programar

Codifica un programa en C en Linux usando el procedimiento `exec()` que tenga la función de un mini-micro shell. Tu mini-micro shell deberá implementar los siguientes comandos:

```
listar (ls)
fecha (date)
renombrar (mv)
borrar (rm)
```

Cuando se ejecute el programa deberá mostrar un prompt:

```
SHELL_SO>$
```

Una vez que el usuario escriba un comando permitido, deberá mostrarse el resultado de la ejecución y deberá mostrar nuevamente el prompt hasta que el usuario escriba un comando de salida, por ejemplo puedes usar la palabra `salir`.

```
SHELL_SO>$ borrar archivo
SHELL_SO>$ salir
```