



PRACTICA 3

🕒 Created	@January 15, 2021 10:31 AM
📅 Date	@October 12, 2020 → October 25, 2020
☰ Tags	
☰ Tema	

Tema i què es fa

Llamadas a sistema básicas. Fork, execlp, waitpid, getpid, etc.

Sessió i codis

▼ myPS.c

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

void error_y_exit(char *msg, int exit_status){
    perror(msg);
    exit(exit_status);
}

void muta_a_PS(char *username){
    execlp("ps", "ps", "-u", username, (char*)NULL);
    error_y_exit("Ha fallado la mutación al ps", 1);
}

int main(int argc, char *argv[]){
    int pid = fork();

    char buffer[80];

    if (argc != 2) error_y_exit("Error, debes especificar al menos un parametro", -1);
    if (pid == -1) error_y_exit("Error del proceso hijo", -1);

    sprintf(buffer, "Soy el proceso: %d\n", getpid());
    write(1,buffer,strlen(buffer));

    if (pid == 0){
```

```

        sprintf(buffer, "Parametros: %s\n", argv[1]);
        write(1,buffer,strlen(buffer));

        muta_a_PS(argv[1]);
    }

    while(1);
}

```

▼ myPS_v0.c

```

#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void error_y_exit(char *msg,int exit_status){
    perror(msg);
    exit(exit_status);
}

int main(int argc, char *argv[]){
    int pid = fork();

    char buffer[80];

    if (argc != 2) error_y_exit("Error, debes especificar al menos un parametro", -1);
    if (pid == -1) error_y_exit("Error del proceso hijo", -1);

    sprintf(buffer, "Soy el proceso: %d\n", getpid());
    write(1,buffer,strlen(buffer));

    if (pid == 0){
        sprintf(buffer, "Parametros: %s\n", argv[1]);
        write(1,buffer,strlen(buffer));
    }

    while(1);
}

```

▼ myPS2.c

```

#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void error_y_exit(char *msg,int exit_status){
    perror(msg);
    exit(exit_status);
}

void muta_a_PS(char *username){
    execlp("ps", "ps", "-u", username, (char*)NULL);
    error_y_exit("Ha fallado la mutación al ps", 1);
}

int main(int argc, char *argv[]){
    char buffer[80];

    if (argc < 2) error_y_exit("Error, debes especificar al menos un parametro", -1);

    sprintf(buffer, "Soy el proceso: %d\n", getpid());
    write(1,buffer,strlen(buffer));
    sprintf(buffer, "Parametros: %s\n", argv[1]);
    write(1,buffer,strlen(buffer));

    for (int i = 1; i < argc; ++i) {
        int pid = fork();

```

```

        if (pid == -1) error_y_exit("Error del proceso hijo", -1);
        else if (pid == 0){
            sprintf(buffer, "Soy el proceso: %d\n", getpid());
            write(1,buffer,strlen(buffer));

            muta_a_PS(argv[i]);
        }
        waitpid(-1, 0, NULL);
    }
}

```

▼ myPS3.c

```

#include <stdio.h>
#include <unistd.h>
#include <string.h>
#include <stdlib.h>

void error_y_exit(char *msg,int exit_status){
    perror(msg);
    exit(exit_status);
}

void muta_a_PS(char *username){
    execlp("ps", "ps", "-u", username, (char*)NULL);
    error_y_exit("Ha fallado la mutación al ps", 1);
}

int main(int argc, char *argv[]){
    char buffer[80];

    if (argc < 2) error_y_exit("Error, debes especificar al menos un parametro", -1);

    sprintf(buffer, "Soy el proceso: %d\n", getpid());
    write(1,buffer,strlen(buffer));
    sprintf(buffer, "Parametros: %s\n", argv[1]);
    write(1,buffer,strlen(buffer));

    for (int i = 1; i < argc; ++i) {
        int pid = fork();
        if (pid == -1) error_y_exit("Error del proceso hijo", -1);
        else if (pid == 0){
            sprintf(buffer, "Soy el proceso: %d\n", getpid());
            write(1,buffer,strlen(buffer));

            muta_a_PS(argv[i]);
        }
    }
    while (waitpid(-1, 0, NULL) > 0);
    char c;
    read(1,&c,sizeof(char));
}

```

▼ parsExec.c

```

#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

void error_y_exit(char *msg,int exit_status) {
    perror(msg);
    exit(exit_status);
}

/* Ejecuta el comando ps -u username mediante la llamada al sistema execlp */
/* Devuelve: el código de error en el caso de que no se haya podido mutar */

int main(int argc,char *argv[])
{

```

```

char buffer[80];
for (int i = 1; i <= 4; ++i) {
    int pid=fork();
    if (pid == -1){ /* Se ha producido un error */
        error_y_exit("Error en el fork", 1);
    //default: /* (pid !=0) && (pid !=-1) */
        /* Escribe aqui el codigo del padre */
    } else if (pid == 0){ /* Escribe aqui el codigo del proceso hijo */
        if (i == 1) execlp("./ListaParametros", "./ListaParametros", "a", "b", (char*)NULL);
        else if (i == 2) execlp("./ListaParametros", "./ListaParametros", (char*)NULL);
        else if (i == 3) execlp("./ListaParametros", "./ListaParametros", "25", "4", (char*)NULL);
        else execlp("./ListaParametros", "./ListaParametros", "muahahaha", "1024", "hola", "adios", (char*)NULL);
        exit(0);
    }
    while ((waitpid(-1, NULL, 0)) > 0);
}

```

Manual

All the info on llamadas a sistema básicas.