Descrição das funções utilizadas no programa do trabalho 1

**pid\_t** → Significa identificação de processo e é usado para representar ids de processo. Sempre que quisermos declarar uma variável que vai lidar com os ids do processo, podemos usar o tipo de dados pid\_t.

O tipo de dados pid\_t é um tipo inteiro com sinal (int com sinal ou podemos dizer int).

**dup2 ()** → torna *newfd* a cópia de *oldfd* , fechando *newfd* primeiro se necessário, mas observe o seguinte:

\*

Se *oldfd* não for um descritor de arquivo válido, a chamada falhará e *newfd* não será fechado.

\*

Se *oldfd* for um descritor de arquivo válido e *newfd* tiver o mesmo valor que *oldfd* , então **dup2** () não fará nada e retornará *newfd* .

**fork()** → Usado para criar um processo filho. é uma função que duplica o processo atual dentro do sistema operacional.Retorno da função, se for:

* negativo: não criou processo filho
* zero: está se executando o processo filho
* positivo: processo pai está sendo executado.

**pipe()** →**pipe** () cria um pipe, um canal de dados unidirecional que pode ser usado para comunicação entre processos.

**execvp()** → executa um arquivo. fornecem um vetor de ponteiros para strings terminadas em '\0' que representam a lista de argumentos disponível ao novo programa. O primeiro argumento, por convenção, deveria apontar para o nome do arquivo associado ao arquivo sendo executado. O vetor de ponteiros deve ser terminado por um ponteiro NULL . Igual à função execv, mas procura na variável de ambiente padrão PATH para encontrar o nome do arquivo se o nome do arquivo não contiver uma barra. recebem como parâmetros o nome do ficheiro executável (também podemos utilizar o *caminho* completo para executar o ficheiro). Se a variável *ficheiro* não contém um caminho, o ficheiro executável vai ser procurado nos directórios especificados na variável ambiente *PATH*.

**waitpid()** → A chamada de sistema **waitpid** () suspende a execução do processo de chamada até que um filho especificado pelo argumento *pid* mude de estado. Por padrão, **waitpid** () espera apenas por filhos encerrados, mas esse comportamento é modificável por meio do argumento *options*.

open() → A função *open ()* estabelece a conexão entre um arquivo e um descritor de arquivo. Ele cria uma descrição de arquivo aberto que se refere a um arquivo e um descritor de arquivo que se refere a essa descrição de arquivo aberto. O descritor de arquivo é usado por outras funções de E / S para se referir a esse arquivo. O argumento de *caminho* aponta para um nome de caminho que nomeia o arquivo.

A função *open ()* retornará um descritor de arquivo para o arquivo nomeado que é o descritor de arquivo mais baixo não aberto atualmente para esse processo. A descrição do arquivo aberto é nova e, portanto, o descritor do arquivo não a compartilha com nenhum outro processo no sistema.

**O\_RDONLY** → Aberto apenas para leitura.

**O\_CREAT →** Se o arquivo existir, este sinalizador não tem efeito, exceto conforme observado em O\_EXCL abaixo. Caso contrário, o arquivo é criado; o ID do usuário do arquivo é definido como o ID do usuário efetivo do processo; o ID do grupo do arquivo é definido como o ID do grupo do diretório pai do arquivo ou como o ID do grupo efetivo do processo;

**O\_RDWR →** Aberto para leitura e escrita. O resultado é indefinido se este sinalizador for aplicado a um FIFO.

**O\_TRUNC →** Se o arquivo existe e é um arquivo normal, e o arquivo é aberto com sucesso O\_RDWR ou O\_WRONLY, seu comprimento é truncado para 0 e o modo e o proprietário não são alterados. Não terá efeito em arquivos especiais FIFO ou arquivos de dispositivos de terminal. Seu efeito em outros tipos de arquivo depende da implementação. O resultado de usar O\_TRUNC com O\_RDONLY é indefinido.

**O\_APPEND →**Se definido, o deslocamento do arquivo será definido para o final do arquivo antes de cada gravação.

**STDIN\_FILENO → pipe\_output. Este é o descritor de arquivo de entrada padrão. Ele é usado para obter entrada do terminal por padrão. As funções scanf () , getc () etc usam o descritor de arquivo stdin para obter as entradas do usuário. O descritor de arquivo stdin também é representado pelo número 0 .**

**STDOUT\_FILENO → terminal\_output. Este é o descritor de arquivo de saída padrão. É usado para imprimir algo no console / terminal por padrão. A função printf () amplamente usada usa stdout para imprimir a saída desejada no console / terminal. O descritor de arquivo stdout também é representado pelo número 1 .**