exec komut ailesini araştıralım.
Farklı kullanım tekniklerini bir program üzerinde gösterelim.

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

#define BUFFER 1024

int main(void) {
    char line[BUFFER];

    while(1) {
        printf("$ ");
        if(!fgets(line, BUFFER, stdin)) break;
        char *p = strchr(line, '\n');
        if (p) *p = 0;
        if(strcmp(line, "exit")==0) break;
        char *args[] = {line, (char*)0};
        int pid= fork();
        if(pid==0) {
            execvp(line, args);
            perror("exec");
            exit(1);
        } else {
            wait(NULL);
        }
    }

    return 0;
}
```

```
int X = open("./output_file.txt", O_WRONLY);

pid_t fk = fork();
if (!fk) {
    dup2(X, 1);

    close(X);

    execl("/bin/echo", "/bin/echo", "hello world");
    _exit(127);
} else if (fk == -1) {

    perror("fork");
}
close(X);
```

```
pid_t parent = getpid();
pid_t pid = fork();
```

```
if (pid == -1)
{

}
else if (pid > 0)
{
    int status;
    waitpid(pid, &status, 0);
}
else
{

    execve(...);
    _exit(EXIT_FAILURE);   // exec never returns
}
```