#include <stdio.h>

#include <sys/types.h>

void parse(char \*line, char \*\*argv)

{

while (\*line != '\0') { /\* if not the end of line ....... \*/

while (\*line == ' ' || \*line == '\t' || \*line == '\n')

\*line++ = '\0'; /\* replace white spaces with 0 \*/

\*argv++ = line; /\* save the argument position \*/

while (\*line != '\0' && \*line != ' ' &&

\*line != '\t' && \*line != '\n')

line++; /\* skip the argument until ... \*/

}

\*argv = '\0'; /\* mark the end of argument list \*/

}

void execute(char \*\*argv)

{

pid\_t pid;

int status;

if ((pid = fork()) < 0) { /\* fork a child process \*/

printf("\*\*\* ERROR: forking child process failed\n");

exit(1);

}

else if (pid == 0) { /\* for the child process: \*/

if (execvp(\*argv, argv) < 0) { /\* execute the command \*/

printf("\*\*\* ERROR: exec failed\n");

exit(1);

}

}

else { /\* for the parent: \*/

while (wait(&status) != pid) /\* wait for completion \*/

;

}

}

void main(void)

{

char line[1024]; /\* the input line \*/

char \*argv[64]; /\* the command line argument \*/

while (1) { /\* repeat until done .... \*/

printf("Shell -> "); /\* display a prompt \*/

gets(line); /\* read in the command line \*/

printf("\n");

parse(line, argv); /\* parse the line \*/

if (strcmp(argv[0], "exit") == 0) /\* is it an "exit"? \*/

exit(0); /\* exit if it is \*/

execute(argv); /\* otherwise, execute the command \*/

}

}