**Assignment Number 01**

**Name:** Mihir Unmesh Patil

**Roll NO:** TYCOC213

**Batch:** C/C-3

**CODE:**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <string.h>

void copy\_file(const char \*source, const char \*destination) {

FILE \*src = fopen(source, "r");

if (!src) {

perror("Error opening source file");

exit(EXIT\_FAILURE);

}

FILE \*dest = fopen(destination, "w");

if (!dest) {

perror("Error opening destination file");

fclose(src);

exit(EXIT\_FAILURE);

}

char buffer[1024];

size\_t bytes;

while ((bytes = fread(buffer, 1, sizeof(buffer), src)) > 0) {

fwrite(buffer, 1, bytes, dest);

}

fclose(src);

fclose(dest);

printf("File copied successfully.\n");

}

void grep\_in\_file(const char \*filename, const char \*search\_term) {

FILE \*file = fopen(filename, "r");

if (!file) {

perror("Error opening file");

exit(EXIT\_FAILURE);

}

char line[1024];

while (fgets(line, sizeof(line), file)) {

if (strstr(line, search\_term)) {

printf("%s", line);

}

}

fclose(file);

}

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

if (argc < 3) {

fprintf(stderr, "Usage: %s <command> <args>\n", argv[0]);

exit(EXIT\_FAILURE);

}

pid\_t pid = fork();

if (pid < 0) {

perror("Fork failed");

exit(EXIT\_FAILURE);

} else if (pid == 0) {

// Child process

printf("Child Process: PID = %d\n", getpid());

if (strcmp(argv[1], "cp") == 0 && argc == 4) {

copy\_file(argv[2], argv[3]);

} else if (strcmp(argv[1], "grep") == 0 && argc == 4) {

grep\_in\_file(argv[3], argv[2]);

} else {

fprintf(stderr, "Invalid command or arguments\n");

}

exit(EXIT\_SUCCESS);

} else {

// Parent process

printf("Parent Process: PID = %d, waiting for child to finish...\n", getpid());

wait(NULL);

printf("Child process finished execution.\n");

}

return 0;

}

**OUTPUT:**

