B.BHANUTEJA REDDY-192325016

25. Construct a C program to implement the I/O system calls of UNIX (fcntl, seek, stat, opendir, readdir)

AIM:

To construct a C program that implements UNIX I/O system calls, including fcntl, lseek, stat, opendir, and readdir.

ALGORITHM:

1. **Initialization:**
   a. Include necessary headers (fcntl.h, unistd.h, sys/stat.h, dirent.h, and stdio.h).
2. **Open File with open():**
   a. Create or open a file with the O_CREAT | O_RDWR flags.
   b. Print the file descriptor.
3. **Duplicate File Descriptor with fcntl():**
   a. Use fcntl() with F_DUPFD to duplicate the file descriptor.
   b. Print the new file descriptor.
4. **Write and Read with write() and read():**
   a. Write data to the file using write().
   b. Use lseek() to reset the file pointer.
   c. Read the file content into a buffer and print it.
5. **Get File Metadata with stat():**
   a. Use stat() to retrieve file metadata (size, permissions).
   b. Print the retrieved details.
6. **Open Directory with opendir():**
   a. Open the current directory using opendir().
   b. Use readdir() to iterate through the directory contents and print the file names.
7. **Close Resources:**
   a. Close the file descriptors and directory stream.

PROCEDURE:

1. **File Descriptor Operations (fcntl):**
   a. Call open() to create/open a file and store the file descriptor.
   b. Use fcntl() to duplicate the file descriptor.

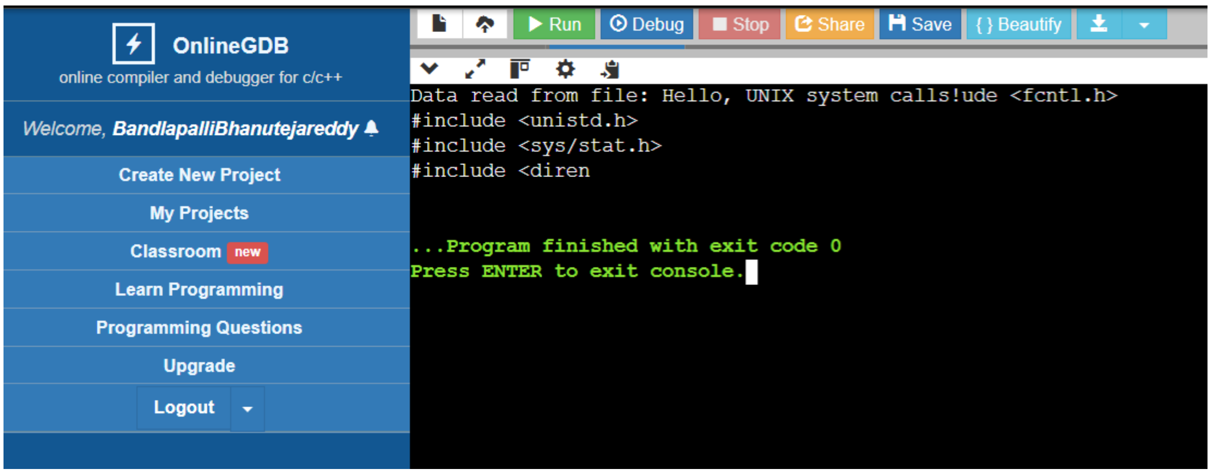
2. **File Operations (write, lseek, read):**
   a. Write data to the file.
   b. Reset the file pointer using lseek().
   c. Read the data from the file and store it in a buffer.
3. **Retrieve File Metadata (stat):**
   a. Use stat() to get the file's size, permissions, and other metadata.
4. **Directory Operations (opendir, readdir):**
   a. Open the current directory using opendir().
   b. Iterate through the directory contents using readdir().
5. **Print Results:**
   a. Print file descriptor details, file content, metadata, and directory contents.
6. **Resource Management:**
   a. Close all opened file descriptors and directory streams.

CODE:

```
#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

#include <sys/stat.h>

#include <dirent.h>


int main() {

  int file = open("example.txt", O_CREAT | O_RDWR, 0644);

  if (file < 0) {

    perror("Error opening file");

    return 1;

  }

  printf("File descriptor: %d\n", file);


  int new_fd = fcntl(file, F_DUPFD, 0);
```

```
printf("Duplicated file descriptor: %d\n", new_fd);


write(file, "Hello, world!", 13);
lseek(file, 0, SEEK_SET);
char buffer[100];
int bytesRead = read(file, buffer, sizeof(buffer) - 1);
if (bytesRead > 0) {
   buffer[bytesRead] = '\0';
   printf("Read from file: %s\n", buffer);
}


struct stat fileStat;
if (stat("example.txt", &fileStat) == 0) {
   printf("File size: %ld bytes\n", fileStat.st_size);
   printf("File permissions: %o\n", fileStat.st_mode & 0777);
} else {
   perror("Error using stat");
}


DIR *dir = opendir(".");
if (dir) {
   printf("Directory contents:\n");
   struct dirent *entry;
   while ((entry = readdir(dir)) != NULL) {
      printf("%s\n", entry->d_name);
   }
   closedir(dir);
} else {
      perror("Error opening directory");
    }


    close(file);

    close(new_fd);

    return 0;

}
```

OUTPUT:



```
Data read from file: Hello, UNIX system calls!ude <fcntl.h>
#include <unistd.h>
#include <sys/stat.h>
#include <diren


...Program finished with exit code 0
Press ENTER to exit console.
```