توضیحات کلی

کد به زبان سی (C) برای پیمایش یک دایرکتوری و انجام عملیات‌هایی مانند شمارش فایل‌ها، شناسایی و حذف فایل‌های تکراری و محاسبه اندازه دایرکتوری‌ها نوشته شده است. این کد از کتابخانه‌های مختلفی برای مدیریت فایل‌ها، دایرکتوری‌ها، حافظه مشترک و همزمانی استفاده می‌کند.

ورودی و خروجی

ورودی

ورودی این کد مسیر دایرکتوری است که توسط کاربر وارد می‌شود .کد از کاربر می‌خواهد تا مسیر دایرکتوری را به عنوان ورودی وارد کند:

```
printf("Enter the root directory path: ");
scanf("%s", rootDirectory);
```

نوع ورودی(char* :مسیر دایرکتوری)

خروجی

خروجی کد شامل اطلاعات زیر است

1. **تعداد کل فایل‌ها**
2. **تعداد هر نوع فایل**
3. **فایل‌های تکراری و مسیرهای آن‌ها**
4. **اندازه دایرکتوری قبل و بعد از حذف فایل‌های تکراری**
5. **اطلاعات مربوط به فایل‌های تکراری در دایرکتوری‌ها**

توابع و عملکرد آن‌ها

1. getFileType

```c
char *getFileType(char *path) {
    char *dot = strrchr(path, '.');
    if (!dot || dot == path) return "";
    return dot + 1;
}
```

عملکرد: گرفتن نوع فایل از مسیر آن با استفاده از پسوند فایل.

2. addFileType

```c
int addFileType(char *fileType) {
    for (int i = 0; i < MAX_FILE_TYPES; i++) {
        if (fileTypes[i] == NULL) {
            fileTypes[i] = strdup(fileType);
            return i;
        } else if (strcmp(fileTypes[i], fileType) == 0) {
            return i;
        }
    }
    return -1;
}
```

عملکرد: اضافه کردن نوع فایل به آرایه‌ای از نوع فایل‌ها و برگرداندن ایندکس آن.

3. traverseDirectory

```c
void traverseDirectory(char *path, int msgqid, int shmid, int processIndex, pthread_mutex_t
*mutex) {
    //...
}
```

عملکرد: پیمایش دایرکتوری و جمع‌آوری اطلاعات فایل‌ها و نوع فایل‌ها.

4. **calculateDirectorySize**

```
void calculateDirectorySize(char *path, off_t *size) {
    //...
}
```

عملکرد: محاسبه اندازه دایرکتوری به صورت بازگشتی.

5. **handleDuplicateFile**

```
void handleDuplicateFiles(char *rootDirectory) {
    //...
}
```

عملکرد: شناسایی و حذف فایل‌های تکراری و نمایش اندازه دایرکتوری قبل و بعد از حذف.

6. **printDuplicateInfo**

```
void printDuplicateInfo(char *rootDirectory) {
    //...
}
```

عملکرد: نمایش اطلاعات فایل‌های تکراری در دایرکتوری‌ها.

کتابخانه‌ها و عملکرد آن‌ها

1. **stdio.h**

توابع استاندارد ورودی/خروجی.

2. **stdlib.h**

توابع استاندارد عمومی مانند تخصیص حافظه و تبدیل داده‌ها.

3. **string.h**

توابع مدیریت رشته‌ها.

4. **stdbool.h**

تعریف نوع داده بولی.

5. **dirent.h**

توابع مدیریت دایرکتوری‌ها.

6. **pthread.h**

توابع مربوط به مدیریت نخ‌ها.

7. **sys/types.h, sys/stat.h, unistd.h**

توابع و ساختارهای مربوط به سیستم فایل.

8. **sys/ipc.h, sys/shm.h, sys/msg.h, sys/wait.h**

توابع و ساختارهای مربوط به حافظه مشترک و پیام‌های بین‌پردازشی.

توضیحات کلی درباره عملکرد کد

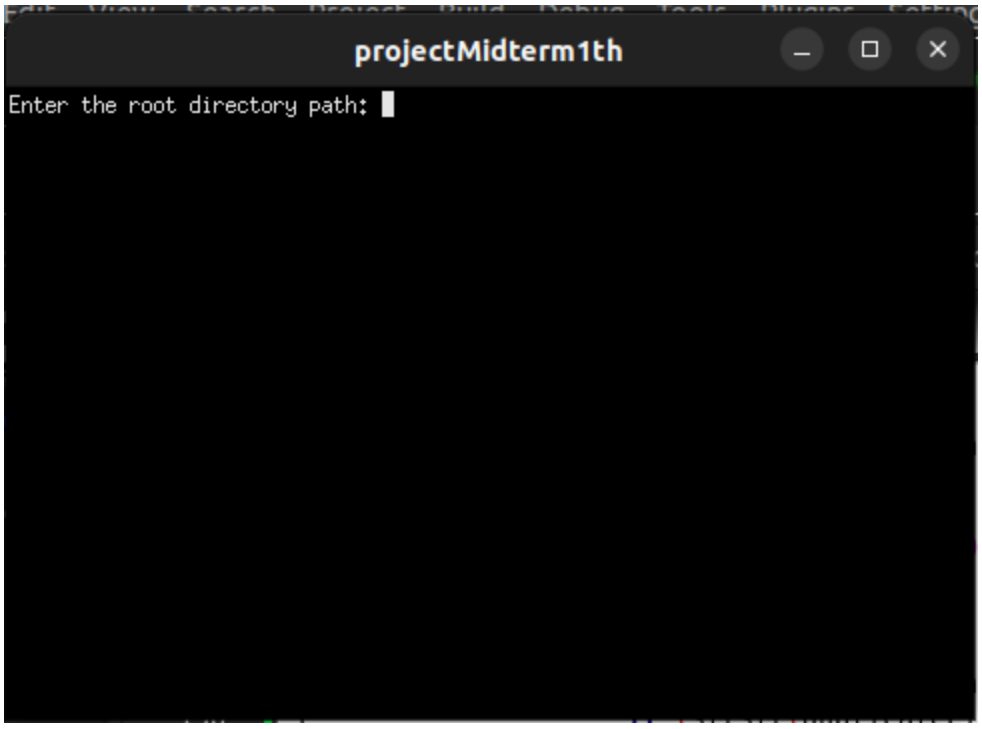
کد یک دایرکتوری و زیر دایرکتوری‌های آن را پیمایش می‌کند، اطلاعات فایل‌ها را جمع‌آوری می‌کند، تعداد هر نوع فایل را شمارش می‌کند، فایل‌های تکراری را شناسایی و حذف می‌کند و در نهایت اطلاعات مربوط به فایل‌ها و دایرکتوری‌ها را نمایش می‌دهد.

این کد می‌تواند در موارد زیر استفاده شود:

1. **تحلیل دایرکتوری‌ها و ساختار فایل‌ها.**
2. **شناسایی و مدیریت فایل‌های تکراری.**
3. **محاسبه فضای استفاده شده در دایرکتوری‌ها.**

نحوه استفاده از کد

برای اجرای این کد، مسیر دایرکتوری مورد نظر خود را به عنوان ورودی وارد کنید. مثلاً:



خروجی مورد انتظار شامل موارد زیر خواهد بود:

1. **تعداد کل فایل‌ها در دایرکتوری و زیر دایرکتوری‌ها.**
2. **تعداد هر نوع فایل) مثلاً.(jpg. ,txt.**

3. **فایل‌های تکراری که شناسایی و حذف شده‌اند.**
4. **اندازه دایرکتوری قبل و بعد از حذف فایل‌های تکراری.**

روند کلی

1. **برنامه از کاربر مسیر دایرکتوری ریشه را دریافت می کند.**
2. **یک حافظه اشتراکی و یک صف پیام ایجاد می شود.**
3. **برای دایرکتوری ریشه، تابع traverseDirectory فراخوانی می شود که به طور بازگشتی در تمام زیردایرکتوری ها گشت می زند.**
4. **برای هر زیردایرکتوری، یک فرآیند جدید ایجاد می شود که traverseDirector را برای آن زیردایرکتوری فراخوانی می کند.**
5. **برای هر فایل معمولی، تابع processFileOrDir فراخوانی می شود که اطلاعات مربوط به فایل را در حافظه اشتراکی ذخیره می کند و تعداد هر نوع فایل را به روز می کند.**
6. **فرآیند اصلی منتظر می ماند تا تمام فرآیندهای فرزند به پایان برسند.**
7. **تابع calculateFileCountAndSize آمار مختلف را از اطلاعات ذخیره شده محاسبه می کند و نتایج را چاپ می کند.**
8. **حافظه اشتراکی و صف پیام آزاد می شوند.**

خروجی

1. **تعداد کل فایل ها**
2. **اندازه کل فایل ها**
3. **نام و اندازه بزرگترین فایل**
4. **نام و اندازه کوچکترین فایل**
5. **تعداد هر نوع فایل (بر اساس پسوند)**

این برنامه از چندین مکانیزم ارتباطات بین فرآیندی (IPC) مانند حافظه اشتراکی و صف پیام استفاده می کند تا اطلاعات را بین فرآیندها به اشتراک بگذارد و هماهنگی را برقرار کند. همچنین از قفل های متقابل (mutex)برای جلوگیری از شرایط نامطلوب در دسترسی همزمان به داده ها استفاده می کند.