

Sistemi Operativi

Laboratorio 2

#include <dirent.h>

Header usato in un programma C che contiene i costrutti per facilitare « l' attraversamento » dei **direttori**.

Alcune funzioni importanti dell' header dirent.h :

Struct dirent

Opendir

Closedir

Readdir

```
#include <stdio.h>
#include <dirent.h>
```

```
int main ()
{
```

```
}
```

Opendir (), dirent (), closedir ()

DIR *opendir (constchar * dirname);

Apre un direttorio in lettura

Valori di ritorno:

Il puntatore al direttorio se corretta

Il puntatore NULL in caso di errore

Struct dirent * readdir (DIR* dirp)

Continua la lettura del direttorio

Valori di ritorno:

Il puntatore al direttorio se corretta

Il puntatore NULL in caso di errore o al termine della lettura

int *closedir (DIR * dirp);

Termina la lettura

Valori di ritorno:

Il valore "0" se corretta

Il valore "-1" in caso di errore

```
#include <stdio.h>
```

```
#include <dirent.h>
```

```
int main ()
```

```
{
```

```
    DIR * dir;
```

```
    dir=opendir(path);
```

```
    struct dirent *dp;
```

```
    dp= readdir (dir);
```

```
    int closedir (DIR *dir);
```

```
}
```

Readdir()

Readdir ()

La funzione readdir () ritorna un puntatore alla struttura dirent che rappresenta l'entry della directory.

La struttura **dirent** contiene un campo definito come **char d_name[]**
Di dimensione non specificato e contenete una stringa con al piu' NAME_MAX caratteri a terminate dal carattere NULL.

La struttura **dirent** contiene un campo definito come **d_type[] :**

DT_DIR (per una cartella)

DT_REG (per un File)

```
#include <stdio.h>
#include <dirent.h>
#include <sys/types.h>

int main ()
{
    DIR * dir;
    dir=opendir(path);
    struct dirent *dp;
    if ((dir = opendir (path) == NULL)
        printf ("opendir () Error");
    else {
        printf ("Content of the root:\n"

    int closedir (DIR *dir);
}
```

Per Stampare la lista delle cartelle:

```
#include <stdio.h>
#include <string.h>
#include <dirent.h>
#include <sys/types.h>

int main ()
{
    DIR * dir;
    dir=opendir(path);
    struct dirent *dp;

    if ((dir = opendir (path) == NULL)
        printf ("opendir () Error");
    else {
        printf ("Content of the root:\n");
        while ((dp = readdir (dir))!= NULL)

            printf ("%s\n", dp -> d_name);
        closedir (dir);
    }
}
```

Contents of roots:

- -
- Dir 1
Dir 2
Dir 3

Rappresenta la cartella corrente

Rappresenta la cartella parent

Per Stampare la lista delle cartelle:

```
#include <stdio.h>
#include <string.h>
#include <dirent.h>
#include <sys/types.h>
```

```
int main ()
{
    DIR * dir;
    dir=opendir(path);
    struct dirent *dp;

    if ((dir = opendir (path) == NULL)
        printf ("opendir () Error");
    else {
        printf ("Content of the root:\n");
        while ((dp = readdir (dir))!= NULL)
            if ((strcmp(dp ->d_name, ".")!=0){
                if ((strcmp(dp ->d_name, "..")!=0){
                    printf ("%s\n", dp -> d_name);
                }
            }
        }
        closedir (dir); }
}
```

Contents of roots:



Dir 1
Dir 2
Dir 3

Rappresenta la cartella corrente

Rappresenta la cartella parent

mkdir (), rmdir ()

int mkdir (constchar * dirp, mode_t mode);

Creano un nuova direttorio

Valori di ritorno:

“0” se ok

“-1” se ok

int rmdir (constchar * dirp);

Cancellano un direttorio (se vuoto)

Valori di ritorno:

“0” se ok

“-1” se ok

```
#include <stdio.h>
```

```
#include <unistd.h>
```

```
#include <sys/stat.h>
```

```
int main ()
```

```
{
```

```
    int mkdir (constchar * dirp, 0777);
```

```
    int rmdir (constchar * dirp);
```

```
}
```

❑ Makefile

- Il **makefile** è un file che contiene:
 - Una struttura di progetto
 - Delle istruzioni per la sua “compilazione”

```
all: compile clean
```

```
compile:
```

```
    gcc -c myprogram.c -o myprogram.o  
    gcc -o myprogram myprogram.o
```

```
clean:
```

```
    rm -f *.o $(EXEC)
```

Esempio di un makefile

- **Make** comando:
 - Legge un makefile
 - Interpreta la struttura del progetto
 - Realizza l'eseguibile

❑ Comandi Principali

1. Creare una directory

mkdir nome della directory

2. Copiare una directory

cp il percorso di origine/nome della directory /il percorso di destinazione

3. Copiare una directory con nuovo nome

cp il percorso di origine / nome della directory /il percorso di destinazione/ Nuova nome della directory

4. Spostare una directory

mv il percorso di origine / nome della directory /il percorso di destinazione/ Nuova nome della directory

5. Rimuovere I file con parte di nome comune

rm il percorso parent del file/*/*la parte comune di nome

6. Rimuovere una directory

rmdir il percorso della directory

❑ Comandi Principali

7. TOUCH

Creare una nuovo file

touch nome di file

Modificare l'ultimo tempo di accesso

touch -a -d "6 hours ago" nome del file

Settare la data dell'ultimo accesso

la -l --time=atime