Operating Systems

Jarosław Koźlak

VMWare Image

- VMWare Player tool: VMWare Player version 12 or newer: https://www.vmware.com/products/workstation-player/workstation-player-evaluation.html

- Image with Linux system: http://jagular.iisg.agh.edu.pl/~kozlak//UbuntuDesktop.tar.gz

Files and directories

- Useful functions:
  open, close, read, write, fcntl, stat, fstat, mkdir, rmdir, opendir, closedir, readdir, rewinddir, ftw, fopen, fclose, fread, fwrite,
- Functions and variables for error handling:
  perror, errno.

- Acces to Unix manual
  man <command/functionname>

open/close

#include <sys/stat.h>

#include <fcntl.h>

int open(const char * pathname, int flags, …, /* mode_t mode*/)

- parameters:
  - pathname
  - flags – a bit mask which specifies the access mode, e.g. O_RDONLY, O_WRONLY, O_RDWR
- returns a file descriptor which is used to refer to the file

- int close(int descriptor)

read

#include <unistd.h>

size_t read(int fd, void *buffer, size_t count)

Example:

```
#define MAX_READ 20
char buffer[MAX_READ]
if(read(STDIN_FILE, buffer,MAX_READ)==-1)
/* error*/
printf(”The input data was”:  %s\n, buffer);
```

write

- ssize_t write(int void *buffer, size_t count)

fopen, fclose

#include <stdio.h>

FILE *fopen(const char *pathname, const char *type)

- type – r/rb, w/wb, a/ab, …

fclose(FILE f)

fread, fwrite,

#include <stdio.h>

size_t fread(void *ptr, size_t size, size_t bobj, FILE * fp)

size_t fwrite(const void *ptr, size_t size, size_t nobj, FILE *fp)

opendir, closedir,rewinddir

```
#include <dirent.h>
DIR *  opendir(const char *pathneme)
int closedir(DIR *dp)
struct dirent *readdier(DIR *dp)
```

Exercise 3 (1)

- **Task 1. System calls and library functions - comparison of efficiency of system calls and library functions**

- Aim: Develop a program which compares efficiency of system calls and library functions for handling I/O operations.

- Description: Write two programs:
  - Task1a which creates a file with the size given as and argument
  - Task1b which copies this file a numer of times specified by one of the arguments using a size of buffer specified by the next argument.

  Coping program should be written in two versions:
  - (i) using read and write system calls
  - (ii) using fread and fwrite library functions
-

Exercise 3 (2)

- Syntax for Task1a: ex3task1a name-of-file-to-be-generated size-of-file
- Syntax for Task1b: ex3task1b path-to-input-file path-to-output-file mode size-of-buffor

- Note: To generate a random charactter a srand() function can be used.

- For both modes measure *real*, *user* and *system times* for following size of buffers: 1, 4,64,256,512,1024, 4096 8192 bytes,
- Adjust the size of generates file which gives you significant values of time.
- The results (measured times) store in the file ex3task1results.txt.
- Add a few sentences at the end of the file explaining obtained results and identified patterns.

Exercise 3 (3)

- **Task2. Browsing of directories**

- Aims:
  - Write a program which browses a specified directory (given as the first argument of the program) and displays all names of files in this directory, their sizes and times of the last access, it should display nested directories as well.
  - Only information about normal files should be displayed, links and special files should be omitted.
- Program should be written in two versions:
  - using *opendir*, *readdir* and *stat* functios
  - using *nftw* and *stat* function

The used version should be chosen according to the second argument.