OS Problem Sheet #1

Problem 1.1

a) How many system calls and how many library calls does executing **/bin/date** produce?

running **strace /bin/date** on ubuntu terminal gave me 48 system calls.
running **ltrace /bin/date** on the ubuntu terminal gave me 42 library calls.

in both cases how many instances of different system or library calls were displayed on the terminal, for example, **fwrite** and **fputc** as library calls and **open** and **read** as system calls.

b) What are the most frequent (top three) library and system calls and what do these calls do?

mmap2(), close(), and **fstat64()** are most frequent system calls.

close() - closes a file descriptor (*the number between the parentheses. example: close(3)*), so it doesn't refer to any file anymore and can be reused.
mmap2() - creates a new mapping in the virtual address space of the calling process. After the **mmap()** call has returned, the file descriptor can be closed immediately without invalidating any mapping.
fstat64() - returns information about the file.

fputc(), fwrite(), and **__freading()** are most frequent library calls.

fputc() - writes a specified character to a specified stream. (*specified in arguments*)
fwrite() - writes data from the array that one of the arguments points to the given stream.
__freading() - returns a nonzero value if the file specified in the argument is read-only or the last operation on that file was read operation, zero otherwise.

Problem 1.2

a) For each of the following system calls, describe a condition that causes it to fail.

One of the conditions that might cause **int open(const char *path, int oflag, ...)** to fail is the following: **path** isn't valid and there is no **O_CREATE** flag in the arguments. It fails because it can't find the file to open and also doesn't have permission to create it.

int close(int fildes) fails if the file descriptor(*fildes*) isn't valid, meaning that it doesn't point to any open processes.

b) What is the value of errno after a system call completed without an error?

As the Linux man page states: "*The value of **errno** is never set to zero by any system call or library function*." but "*A function that succeeds is allowed to change **errno***". So the value of **errno** should stay the same after a successful system call as it was before the system call was called, unless there is something specified in the system call to change the **errno.**