1. Investigate the man pages of open, read, write and close systems calls. Note the function prototypes and the significance of each argument.

**1. open System Call:**

**Man Page:**

$ man open

**Function Prototype:**

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

* int open(const char \*pathname, int flags);
* int open(const char \*pathname, int flags, mode\_t mode);

**Arguments:**

* **pathname**: The path of the file to be opened.
* **flags**: The access mode of the file (read-only, write-only, read/write, etc.), and additional options (e.g., creation flags).
* **mode** (optional): The file permissions to be set if the file is created. It is only used when **O\_CREAT** flag is present in **flags**.

**2. read System Call:**

**Man Page:**

$ man read

**Function Prototype:**

#include <unistd.h>

ssize\_t read(int fd, void \*buf, size\_t count);

**Arguments:**

* **fd**: File descriptor of the file or socket from which to read.
* **buf**: Buffer where the data will be read into.
* **count**: Number of bytes to read.

**Return Value:**

* On success, the number of bytes read is returned.
* On error, -1 is returned, and **errno** is set to indicate the error.

**3. write System Call:**

**Man Page:**

$ man write

**Function Prototype:**

#include <unistd.h>

* ssize\_t write(int fd, const void \*buf, size\_t count);

**Arguments:**

* **fd**: File descriptor of the file or socket to which to write.
* **buf**: Buffer containing the data to be written.
* **count**: Number of bytes to write.

**Return Value:**

* On success, the number of bytes written is returned.
* On error, -1 is returned, and **errno** is set to indicate the error.

**4. close System Call:**

**Man Page:**

$ man close

**Function Prototype:**

#include <unistd.h>

* int close(int fd);

**Arguments:**

* **fd**: File descriptor to be closed.

**Return Value:**

* On success, 0 is returned.
* On error, -1 is returned, and **errno** is set to indicate the error.