Add system_call

try_hello ⇒ user_land code

1] ADD in Makfile in uprogs

Inside tryhello.c

```
#include "types.h"
#include "stat.h"
#include "user.h"
#include "fs.h"

int
main(int argc, char *argv[])
{
hello();
exit();
}
```

2] user.h ⇒ all the wrappers of system_call and add int hello(void), this is nothing but prototype

> 💡 :e filename

3] usys.S ⇒ assembly code file , it is a macro which calls syscall

Now, adding the system_call

4] in sysfile.c add the main code for system_call

```
int sys_hello(void)
{
cprintf("hello\n");
return 0;
}
```

5] Add SYS_hello in the syscall.c file

6] then add SYS-hello in syscall.h

trylseeksys

SYS_lseek 23

```
int sys_lseek(void){
int fd;
int offset;
int whence;
struct file *f;
```

```
// In sysfile.c, add the following function:
int sys_lseek(void) {
int fd;
int offset;
int whence;
struct file *f;
if (argfd(0,0, &fd) < 0 || argint(1, &offset) < 0 || argint(2, &whence) < 0)
return -1;
if (fd < 0 || fd >= NOFILE || (f = myproc()→ofile[fd]) == 0)
return -1;
if (whence == SEEK_SET) {
f→off = offset;
} else if (whence == SEEK_CUR) {
f→off += offset;
} else if (whence == SEEK_END) {
```

```
if (f→ip→type == T_DEV) {
// For devices, we don't support SEEK_END.
return -1;
}
f→off = f→ip→size + offset;
} else {
return -1;
}

return f→off;
}
```

// In sysfile.c, add the following entry to the `syscalls` array:

```
[SYS_lseek]   sys_lseek,
```

// In user.h, add the following system call definition:

```
int lseek(int fd, int offset, int whence);
```

// In usys.S, add the following system call number:

```
#define SYS_lseek  22
```

// In ulib.c, add the following wrapper function:

```
int lseek(int fd, int offset, int whence) {
return syscall(SYS_lseek, fd, offset, whence);
}


int
sys_lseek(void)
{
int fd;
int offset;
int whence;
struct file *file;
if(argfd(0,&fd,&file) <0 || argint(1,&offset)<0 || argint(2,&whence) <0 )
return -1;
if(whence ==0 )
file→off=offset;
```

else if(whence == 1)
file→off=file→off+offset;
else
file→off=file→ip→size+offset;

```
    return file->off;
```

}

#include "types.h"
#include "stat.h"
#include "user.h"
#include "fcntl.h"
#define SEEK_SET 0
#define SEEK_CUR 1
#define SEEK_END 2
int main() {
int fd, offset, whence;

```
// Open a file
fd = open("testfile.txt", O_RDWR | O_CREATE);
if (fd < 0) {
    printf(2, "Error: Cannot open or create file\\n");
    exit();
}

// Write some content to the file
write(fd, "Hello, XV6!", 12);

// Seek to the beginning of the file
offset = 0;
whence = SEEK_SET;
int new_offset = lseek(fd, offset, whence);
printf(1, "Seek to the beginning. New offset: %d\\n", ne>
// Seek 5 bytes forward from the current offset
offset = 5;
```

offset = 5;
whence = SEEK_CUR;
new_offset = lseek(fd, offset, whence);
printf(1, "Seek 5 bytes forward. New offset: %d\n", new>

```
// Seek to the end of the file
offset = 0;
whence = SEEK_END;
new_offset = lseek(fd, offset, whence);
printf(1, "Seek to the end. New offset: %d\\n", new_offs>

// Close the file
close(fd);

exit();
```

}