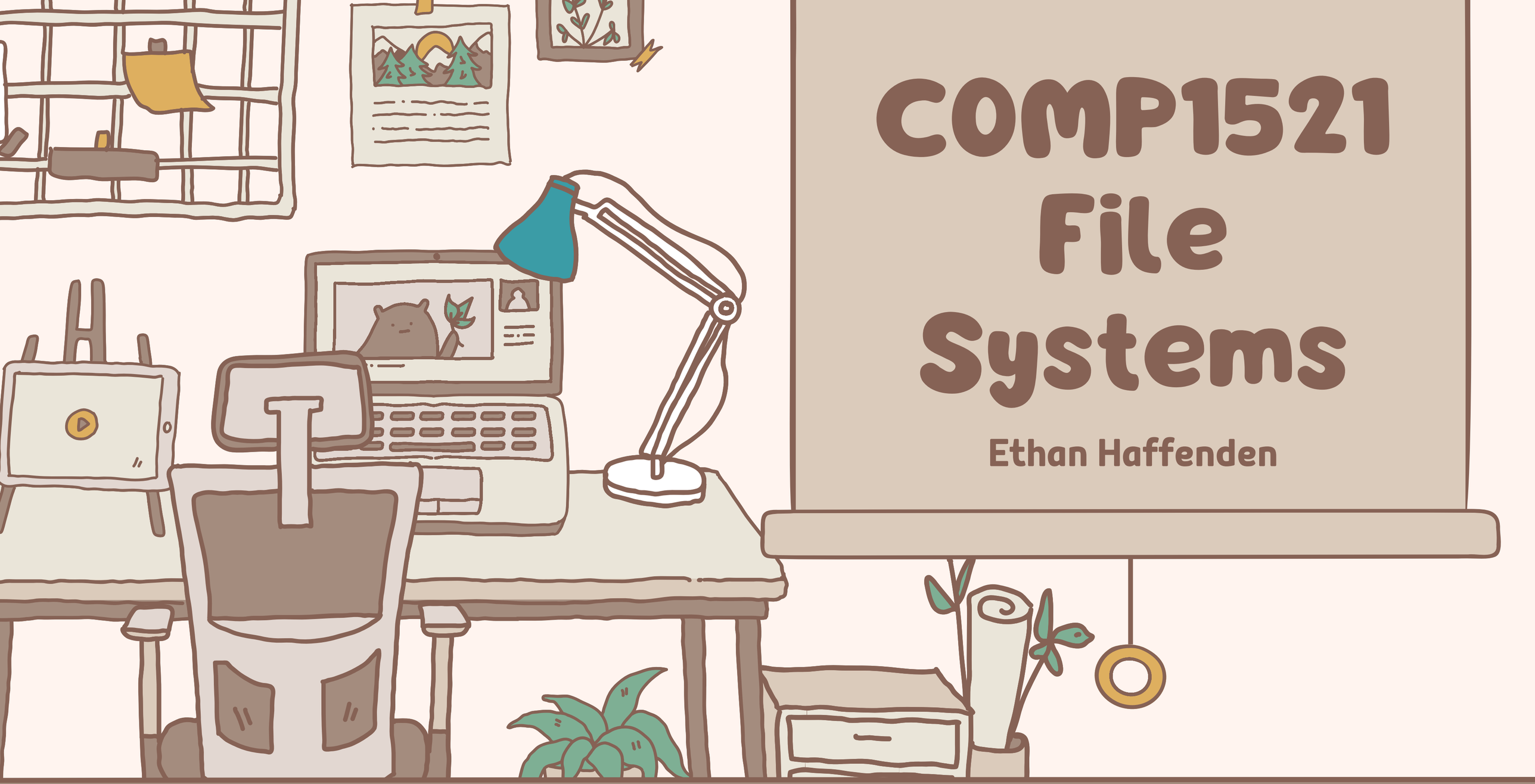
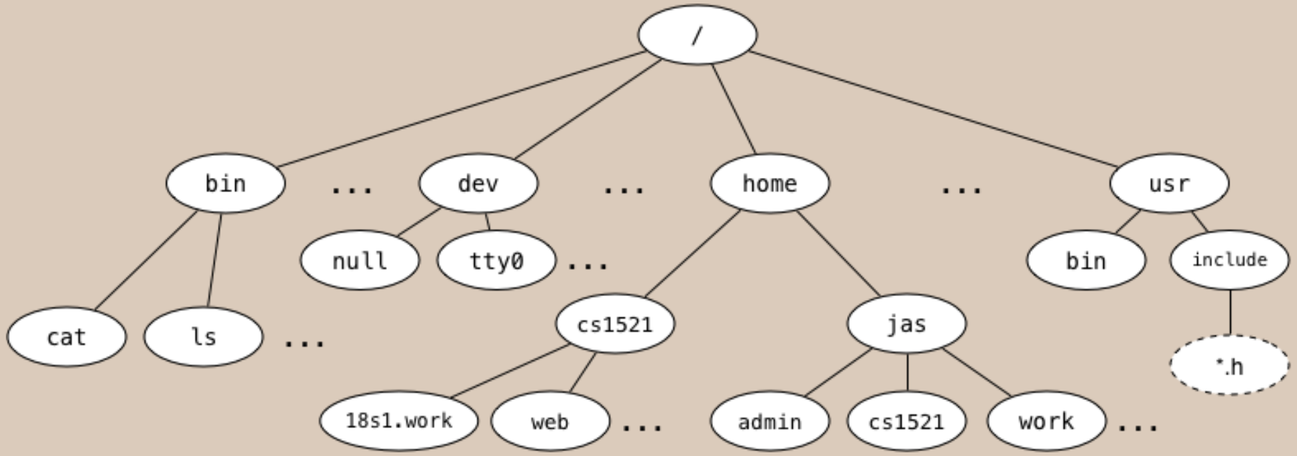


COMP1521 File Systems

Ethan Haffenden

Question 1



- What is the full pathname of COMP1521's web directory?
- Which directory is ~jas/../..?
- Links to the children of a given directory are stored as entries in the directory structure. Where is the link to the parent directory stored?
- What kind of filesystem object is cat?
- what kind of filesystem object is home?
- What kind of filesystem object is tty0?
- What kind of filesystem is a symbolic link? What value dues it contain?
- Symbolic links change the filesystem from a tree structure to a graph structure. How do they do this?

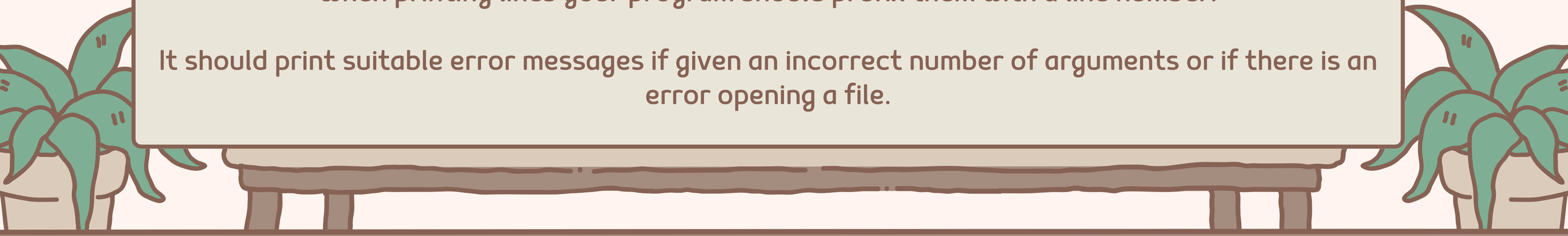
Question 2

Write a C program, fgrep.c, which is given 1+ command-line arguments which is a string to search for.
If there is only 1 command-line argument it should read lines from stdin and print them to stdout iff they contain the string specified as the first command line argument.

If there are 2 or more command line arguments, it should treat arguments after the first as filenames and print any lines they contain which contain the string specified as the first command line arguments.

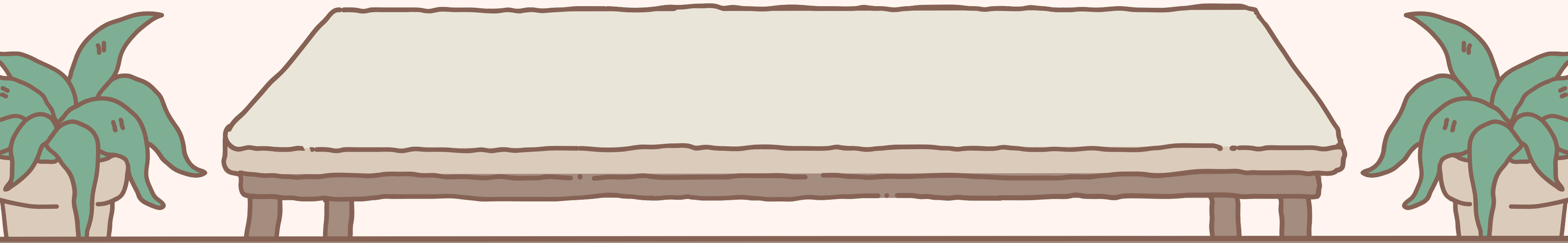
When printing lines your program should prefix them with a line number.

It should print suitable error messages if given an incorrect number of arguments or if there is an error opening a file.



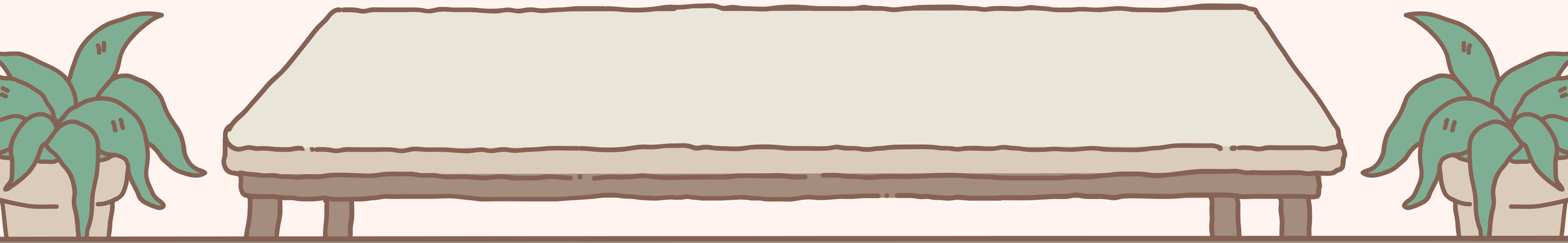
Question 3

WHAT DOES FOPEN DO? WHAT ARE ITS PARAMETERS?



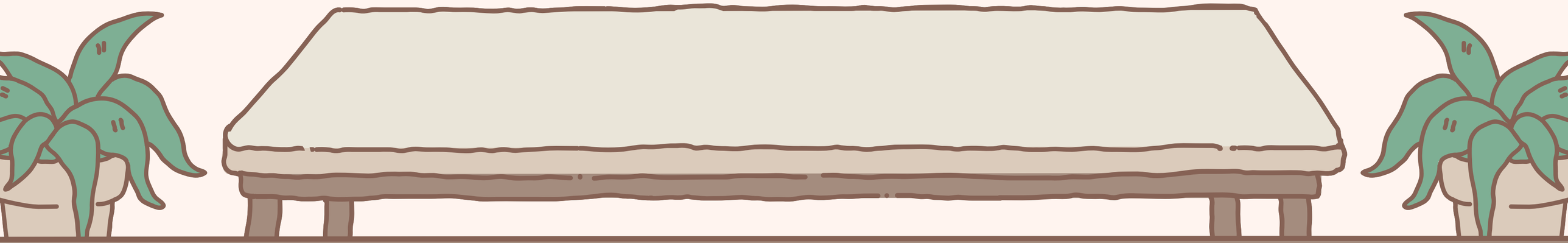
Question 4

WHAT ARE SOME CIRCUMSTANCES WHEN FOPEN RETURNS NULL?



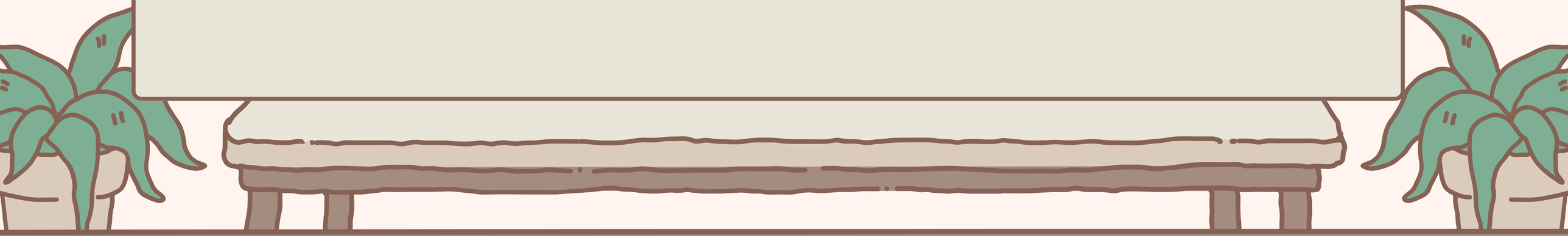
Question 5

HOW DO YOU PRINT THE SPECIFIC REASON THAT CAUSED FOPEN TO RETURN NULL?



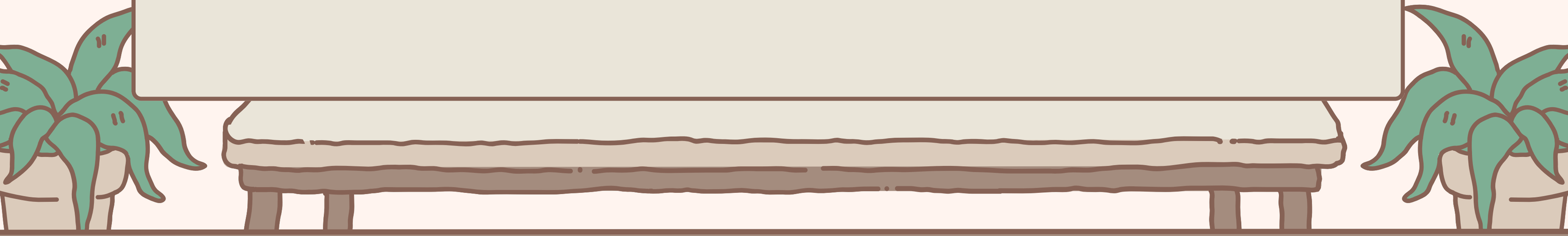
Question 6

Write a C program, first_line.c, which is given one command-line argument, the name of a file, and which prints the first line of that file to stdout. If given an incorrect number of arguments, or if there was an error opening the file, it should print a suitable error message.



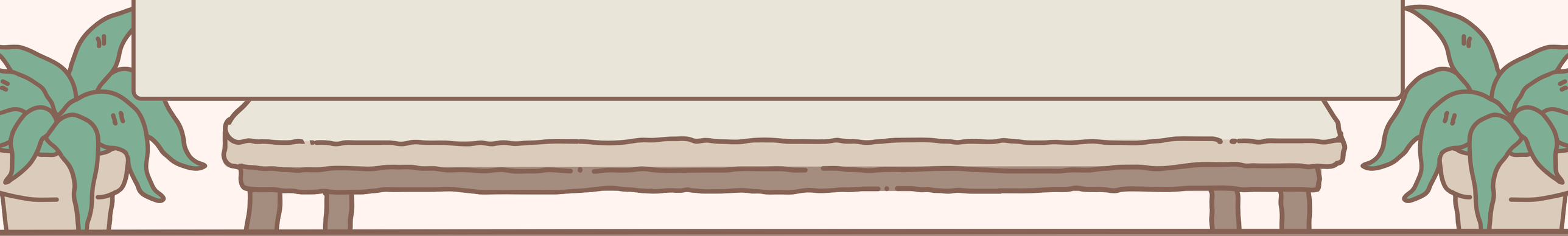
Question 7

Write a C program, write_line.c, which is given one command-line argument, the name of a file, and which reads a line from stdin, and writes it to the specified file; if the file exists, it should be overwritten.



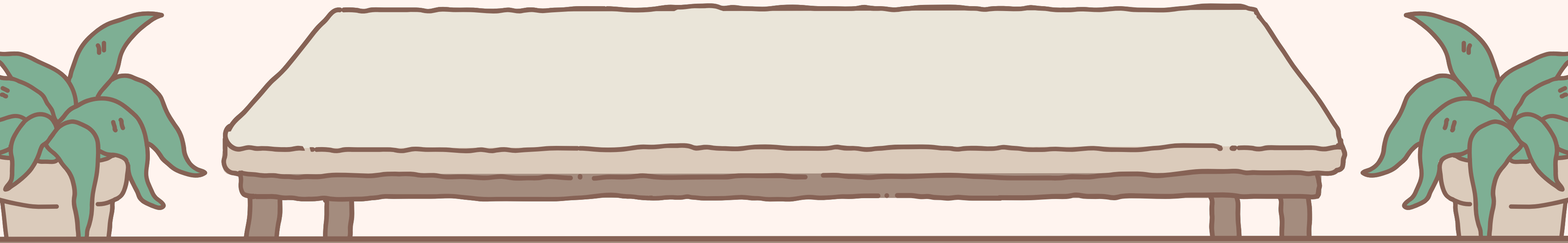
Question 8

Write a C program, append_line.c, which is given one command-line argument, the name of a file, and which reads a line from stdin and appends it to the specified file.



Question 9

WHY SHOULD YOU NOT USE FGETS OR FPUTS WITH BINARY DATA?

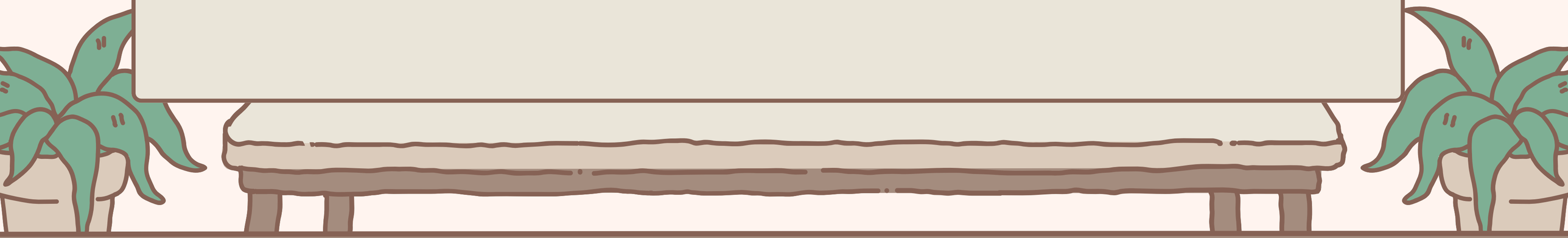


Question 10

What does the following printf statement display?

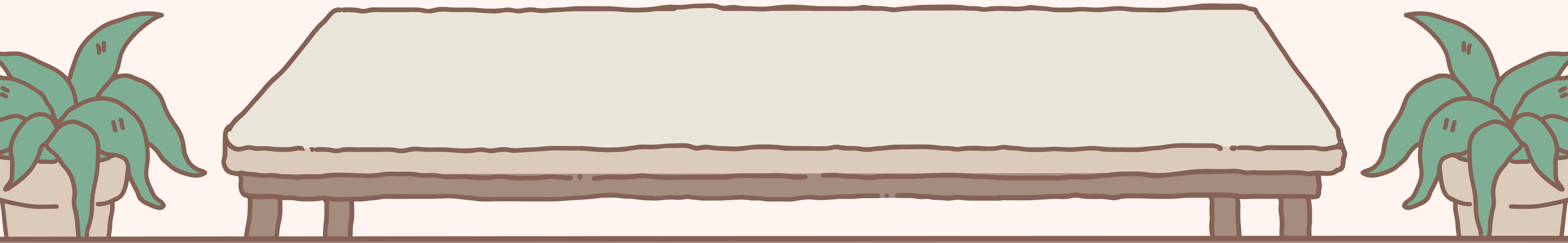
```
PRINTF ("%C%C%C%C%C%C", 72, 101, 0X6C, 108, 111, 0X0A);
```

Try to work it out without compiling and running the code. The ascii manual page will help with this read it by running “man 7 ascii”. Then, check your answer by compiling and running.



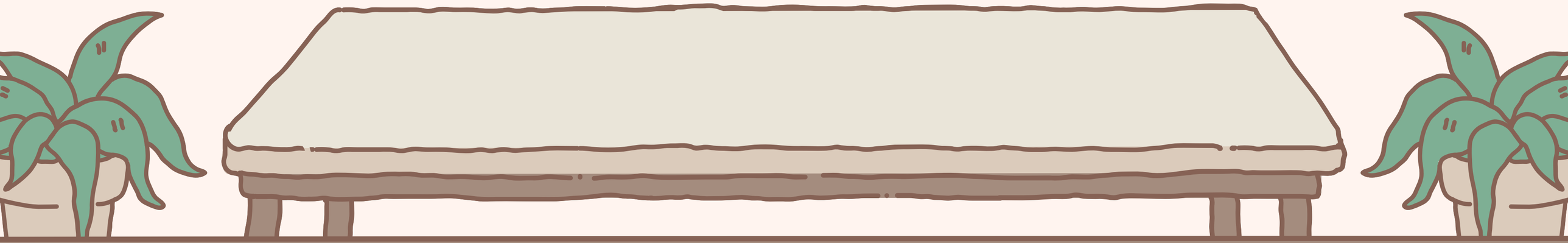
Question 11

HOW MANY DIFFERENT VALUES CAN FGETC RETURN?



Question 12

WHY ARE THE BNAMES OF FGTEC, FPUTC, GETC, PUTC, PUTCHAR, AND GETCHAR MISLEADING?

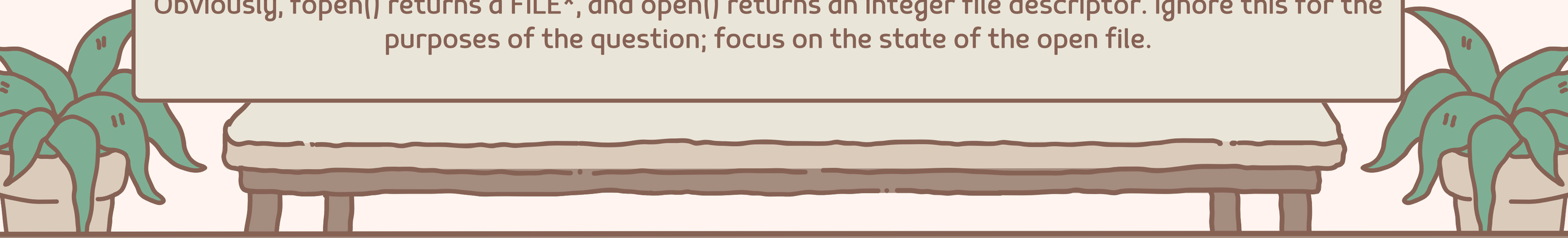


Question 13

For each of the following calls to the fopen() library function, give an open() system call that has equivalent semantics relative to the state of the file.

fopen(FilePath, "r")
fopen(FilePath, "a")
fopen(FilePath, "w")
fopen(FilePath, "r+")
fopen(FilePath, "w+")

Obviously, fopen() returns a FILE*, and open() returns an integer file descriptor. Ignore this for the purposes of the question; focus on the state of the open file.



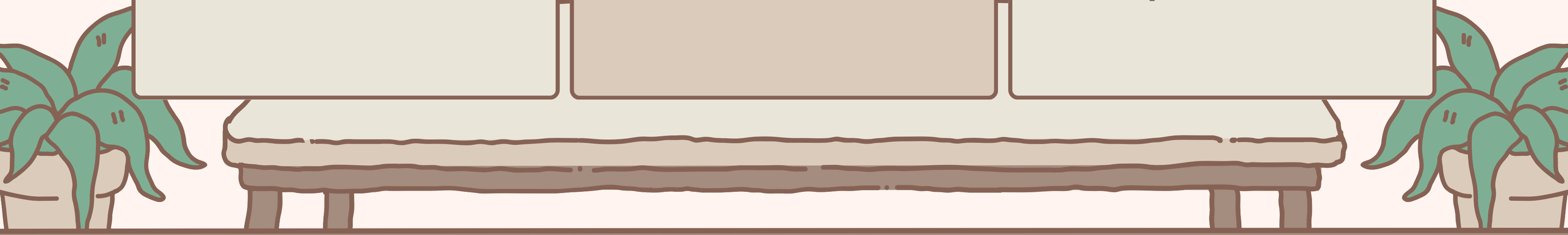
Question 14

CONSIDER LSEEK(FD, OFFSET, WHENCE) FUNCTION

What is its purpose?

When would it be useful?

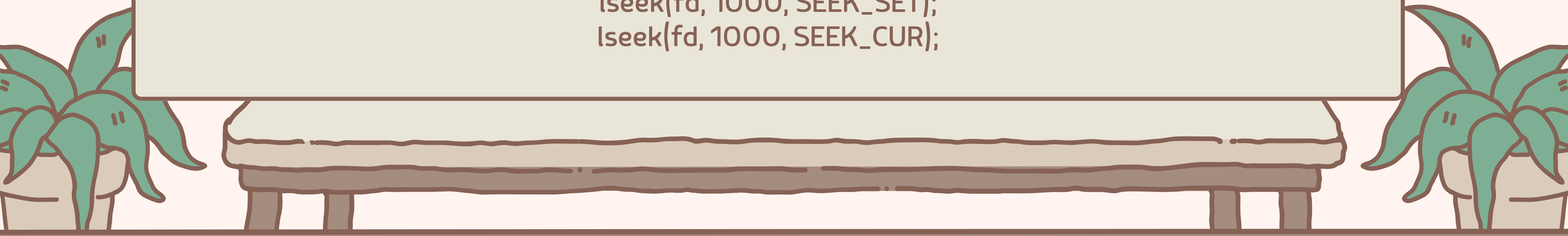
what does its return value represent?



Question 15

Consider a file of size 10000 bytes, open for reading on file descriptor fd, initially positioned at the start of the file (offset 0). What will be the file position after each of these calls to lseek()? Assume that they are executed in sequence, and one will change the file state that the next one deals with.

```
lseek(fd, 0, SEEK_END);
lseek(fd, -1000, SEEK_CUR);
lseek(fd, 0, SEEK_SET);
lseek(fd, -100, SEEK_SET);
lseek(fd, 1000, SEEK_SET);
lseek(fd, 1000, SEEK_CUR);
```



Question 16

If a file xyz contains 2500 bytes, and it is scanned using the following code

Assume that all of the relevant #include's are done.

How many calls with be made to the read() function, and what is the value of nb after each call?

```
int fd;          // open file descriptor
int nb;          // # bytes read
int ns = 0;      // # spaces
char buf[BUFSIZ]; // input buffer

fd = open ("xyz", O_RDONLY);
assert (fd >= 0);
while ((nb = read (fd, buf, 1000)) > 0) {
    for (int i = 0; i < nb; i++)
        if (isspace (buf[i]))
            ns++;
}
close (fd);
```