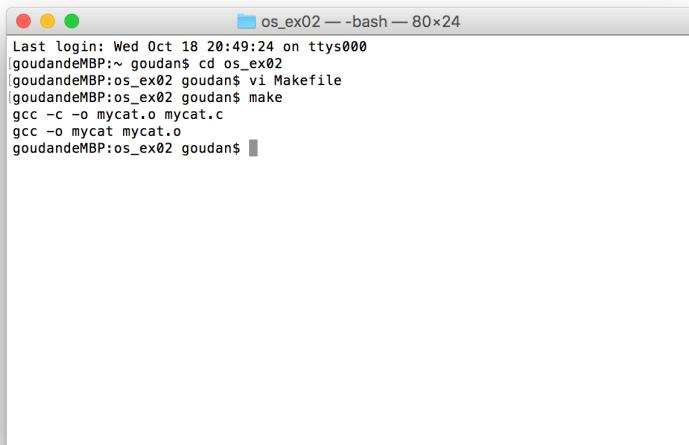


EX1:

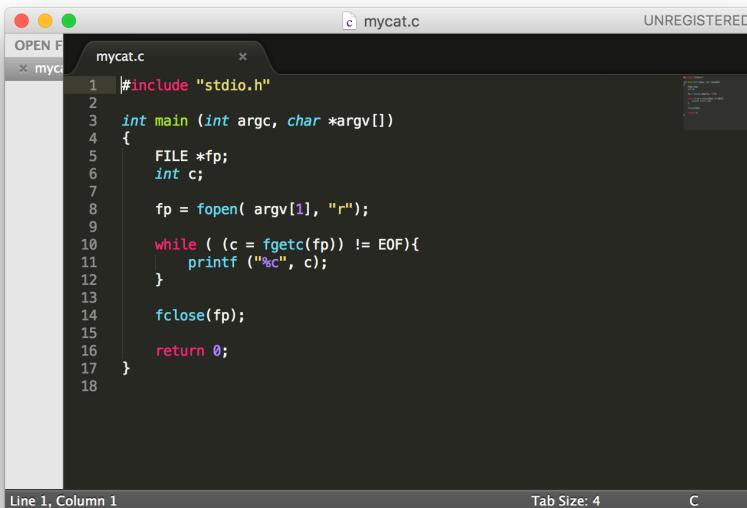
1.



A screenshot of a Mac OS X terminal window titled "os_ex02 — bash — 80x24". The window shows the following command history:

```
Last login: Wed Oct 18 20:49:24 on ttys000
goudandeMBP:~ goudan$ cd os_ex02
goudandeMBP:os_ex02 goudan$ vi Makefile
goudandeMBP:os_ex02 goudan$ make
gcc -c -o mycat.o mycat.c
gcc -o mycat mycat.o
goudandeMBP:os_ex02 goudan$
```

2.

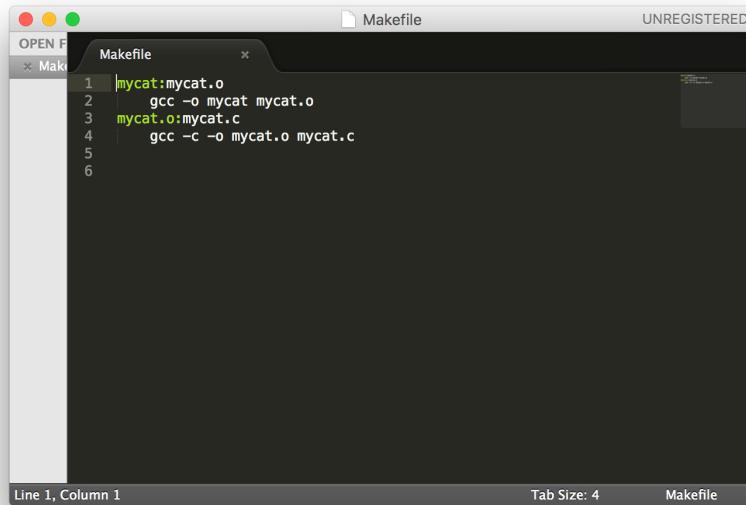


A screenshot of a code editor window titled "mycat.c". The file contains the following C code:

```
#include <stdio.h>
int main (int argc, char *argv[])
{
    FILE *fp;
    int c;
    fp = fopen( argv[1], "r" );
    while ( (c = fgetc(fp)) != EOF){
        printf ("%c", c);
    }
    fclose(fp);
    return 0;
}
```

The status bar at the bottom indicates "Line 1, Column 1" and "Tab Size: 4".

3.

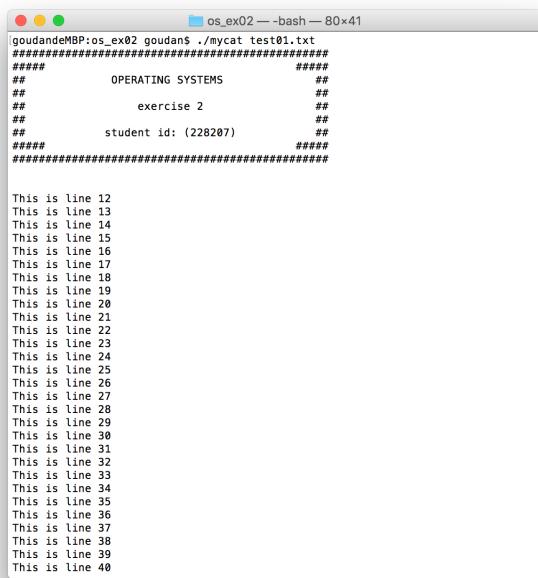


A screenshot of a terminal window titled "Makefile". The window shows a single tab labeled "Makefile". The content of the file is:

```
1 |mycat:mycat.o
2 |    gcc -o mycat mycat.o
3 |mycat.o:mycat.c
4 |    gcc -c -o mycat.o mycat.c
5 |
6 |
```

The terminal status bar at the bottom indicates "Line 1, Column 1" and "Tab Size: 4".

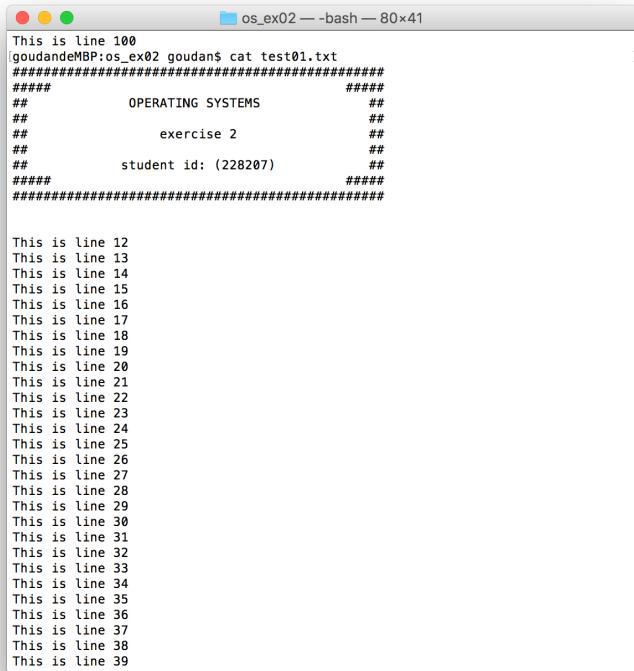
mycat:



A screenshot of a terminal window titled "os_ex02 — bash — 80x41". The command entered is "goudandeMBP:os_ex02 goudans ./mycat test01.txt". The output of the program is:

```
#####
##### OPERATING SYSTEMS #####
##          exercise 2      ##
##          student id: (228207)  ##
#####
This is line 12
This is line 13
This is line 14
This is line 15
This is line 16
This is line 17
This is line 18
This is line 19
This is line 20
This is line 21
This is line 22
This is line 23
This is line 24
This is line 25
This is line 26
This is line 27
This is line 28
This is line 29
This is line 30
This is line 31
This is line 32
This is line 33
This is line 34
This is line 35
This is line 36
This is line 37
This is line 38
This is line 39
This is line 40
```

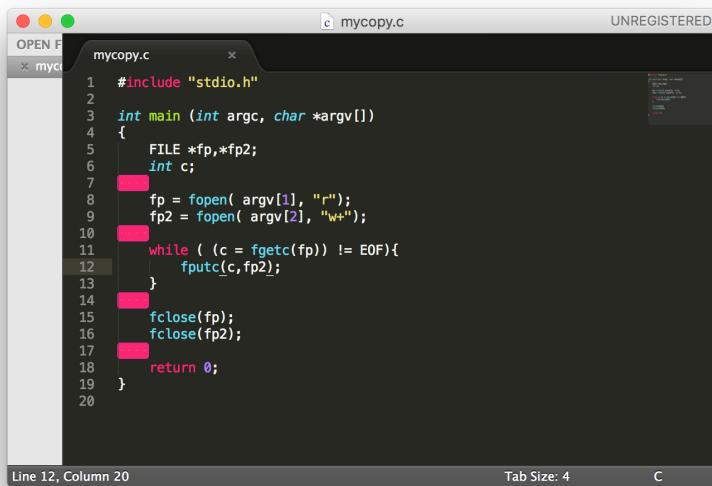
cat:



```
This is line 100
goudandeMBP:os_ex02 goudans$ cat test01.txt
#####
##          OPERATING SYSTEMS      ##
##          exercise 2           ##
##          student id: (228207)   ##
#####

This is line 12
This is line 13
This is line 14
This is line 15
This is line 16
This is line 17
This is line 18
This is line 19
This is line 20
This is line 21
This is line 22
This is line 23
This is line 24
This is line 25
This is line 26
This is line 27
This is line 28
This is line 29
This is line 30
This is line 31
This is line 32
This is line 33
This is line 34
This is line 35
This is line 36
This is line 37
This is line 38
This is line 39
```

4.



```
mycopy.c          UNREGISTERED
OPEN FILE
mycopy.c
1 #include <stdio.h>
2
3 int main (int argc, char *argv[])
4 {
5     FILE *fp,*fp2;
6     int c;
7
8     fp = fopen( argv[1], "r" );
9     fp2 = fopen( argv[2], "w+" );
10
11    while ( (c = fgetc(fp)) != EOF){
12        fputc(c,fp2);
13    }
14
15    fclose(fp);
16    fclose(fp2);
17
18    return 0;
19 }
20

Line 12, Column 20
Tab Size: 4
C
```

```
[goudandeMBP:os_ex02 goudan$ make
gcc -c -o mycopy.o mycopy.c
gcc -o mycopy mycopy.o
[goudandeMBP:os_ex02 goudan$ ./mycopy test01.txt copy_test01.txt
[goudandeMBP:os_ex02 goudan$ ls
Makefile      mycat          mycat.o        mycopy.c      test01.txt
copy_test01.txt mycat.c       mycopy         mycopy.o
goudandeMBP:os_ex02 goudan$ ]]
```

5.

A stream opened in text mode we get newline translation on non-*nix systems (it's also used for network communications, but this isn't supported by the standard library). In *nix newline is just ASCII linefeed, \n, both for internal and external representation of text. In Windows the external representation often uses a carriage return + linefeed pair, "CRLF" (ASCII codes 13 and 10), which is converted to a single \n on input, and conversely on output.

EX2

A screenshot of a terminal window titled "Makefile". The window shows a Makefile with the following content:

```
1 BUILD_DIR = bin
2
3 mycopy:mycopy.o
4     gcc -c -o mycopy mycopy.o
5 mycopy.o:mycopy.c
6     gcc -c -o mycopy.o mycopy.c
7
8 .PHONY: install
9 install:
10    mkdir -p $(BUILD_DIR)
11    cp -p mycopy $(BUILD_DIR)
12
13
14 .PHONY: clean
15 clean:
16    rm -f mycopy
17
18
19 .PHONY: distclean
20 distclean:
21    rm -rf mycopy bin
22
23
```

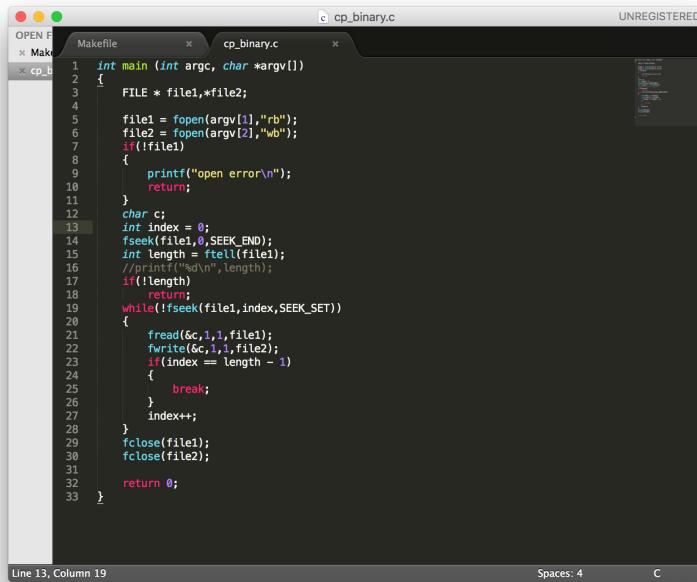
The terminal window also displays the line number "Line 19, Column 15" and the tab size "Tab Size: 4".

A screenshot of a terminal window titled "os_ex02 — bash — 80x22". The window shows the execution of the Makefile commands:

```
[goudandeMBP:os_ex02 goudan$ make
gcc -c -o mycopy.o mycopy.c
gcc -o mycopy mycopy.o
[goudandeMBP:os_ex02 goudan$ ls
Makefile      mycopy      mycopy.c      mycopy.o      test01.txt
[goudandeMBP:os_ex02 goudan$ make install
mkdir -p bin
cp -p mycopy bin
[goudandeMBP:os_ex02 goudan$ ls
Makefile      mycopy      mycopy.o
bin          mycopy.c      test01.txt
[goudandeMBP:os_ex02 goudan$ ls bin
mycopy
[goudandeMBP:os_ex02 goudan$ make clean
rm -f mycopy
[goudandeMBP:os_ex02 goudan$ ls
Makefile      bin          mycopy.c      mycopy.o      test01.txt
[goudandeMBP:os_ex02 goudan$ make distclean
rm -rf mycopy bin
[goudandeMBP:os_ex02 goudan$ ls
Makefile      mycopy.c      mycopy.o      test01.txt
goudandeMBP:os_ex02 goudan$ ]]
```

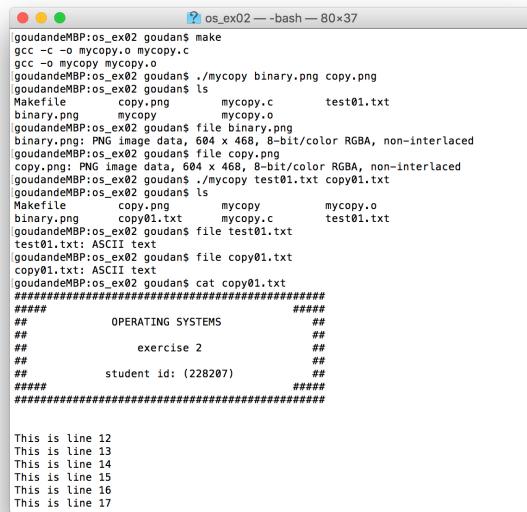
EX3

1.



A screenshot of a code editor window titled "cp_binary.c". The code is a C program named "cp_binary.c" which copies binary data from one file to another. It includes a makefile at the top. The code uses FILE pointers for reading and writing binary files. It checks if both files exist, reads the source file until the end, and writes the data to the destination file. It handles errors by printing an error message and returning. The code editor shows syntax highlighting for C and has tabs for "Makefile" and "cp_binary.c". The status bar at the bottom indicates "Line 13, Column 19", "Spaces: 4", and "C".

```
OPEN FILE cp_binary.c UNREGISTERED
Makefile cp_binary.c
cp_b
1 int main (int argc, char *argv[])
2 {
3     FILE * file1,*file2;
4
5     file1 = fopen(argv[1],"rb");
6     file2 = fopen(argv[2],"wb");
7     if(!file1)
8     {
9         printf("open error\n");
10        return;
11    }
12    char c;
13    int index = 0;
14    fseek(file1,0,SEEK_END);
15    int length = ftell(file1);
16    //printf("%d\n",length);
17    if(!length)
18        return;
19    while(fseek(file1,index,SEEK_SET))
20    {
21        fread(&c,1,file1);
22        fwrite(&c,1,file2);
23        if(index == length - 1)
24        {
25            break;
26        }
27        index++;
28    }
29    fclose(file1);
30    fclose(file2);
31
32    return 0;
33 }
```

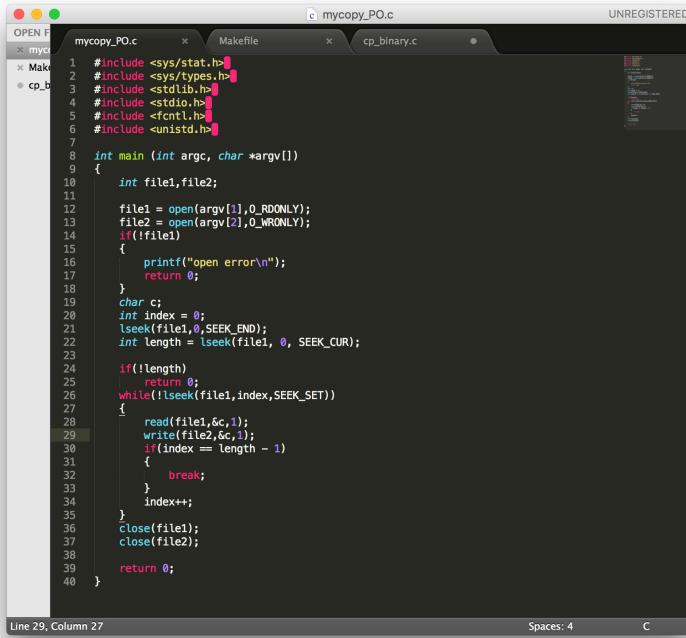


A screenshot of a terminal window titled "os_ex02 -- bash -- 80x37". The terminal shows the execution of a C program named "mycopy". The user first runs "make" to build the program. Then they run "mycopy binary.png copy.png" to copy a PNG image. They then run "ls" to list files, showing "copy.png" and "mycopy.o". Next, they run "copy.png" to copy the image again. They then run "ls" again, showing "copy.png" and "mycopy.o" again. Finally, they run "cat copy01.txt" to view its contents, which are lines 12 through 17 of the original image file.

```
goudandMBP:os_ex02 goudan$ make
gcc -c -o mycopy.o mycopy.c
gcc -o mycopy mycopy.o
goudandMBP:os_ex02 goudan$ ./mycopy binary.png copy.png
goudandMBP:os_ex02 goudan$ ls
Makefile      copy.png      mycopy.c      test01.txt
binary.png   mycopy      mycopy.o
goudandMBP:os_ex02 goudan$ file copy.png
binary.png: PNG image data, 604 x 468, 8-bit/color RGBA, non-interlaced
goudandMBP:os_ex02 goudan$ file copy.png
copy.png: PNG image data, 604 x 468, 8-bit/color RGBA, non-interlaced
goudandMBP:os_ex02 goudan$ ./mycopy test01.txt copy01.txt
goudandMBP:os_ex02 goudan$ ls
Makefile      copy.png      mycopy      mycopy.o
binary.png   copy01.txt   mycopy.c      test01.txt
goudandMBP:os_ex02 goudan$ file test01.txt
test01.txt: ASCII text
goudandMBP:os_ex02 goudan$ file copy01.txt
copy01.txt: ASCII text
goudandMBP:os_ex02 goudan$ cat copy01.txt
#####
#####
##          OPERATING SYSTEMS      ##
##          exercise 2             ##
##          student id: (228207)    ##
#####
#####
This is line 12
This is line 13
This is line 14
This is line 15
This is line 16
This is line 17
```

yes, it's possible to use this program to copy text files.

2.

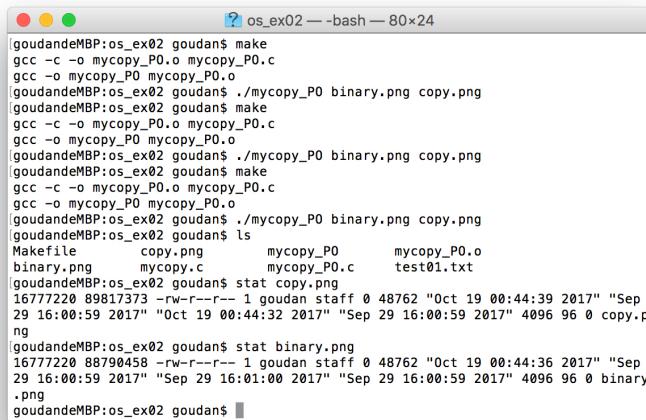


A screenshot of a Mac OS X terminal window titled "mycopy_P0.c". The window contains the C code for a file copy program. The code includes headers for sys/stat.h, sys/types.h, stdlib.h, stdio.h, cntl.h, and unistd.h. It defines a main function that opens two files, reads from the first, and writes to the second until EOF. The code uses lseek to determine the length of the file and read/write operations to copy the contents. The terminal shows the code being typed in, with line numbers 1 through 40 visible at the bottom.

```
#include <sys/stat.h>
#include <sys/types.h>
#include <stdlib.h>
#include <stdio.h>
#include <cntl.h>
#include <unistd.h>

int main (int argc, char *argv[])
{
    int file1,file2;
    file1 = open(argv[1],O_RDONLY);
    file2 = open(argv[2],O_WRONLY);
    if(!file1)
    {
        printf("open error\n");
        return 0;
    }
    char c;
    int index = 0;
    lseek(file1,0,SEEK_END);
    int length = lseek(file1, 0, SEEK_CUR);
    if(!length)
        return 0;
    while(lseek(file1,index,SEEK_SET))
    {
        read(file1,&c,1);
        write(file2,&c,1);
        if(index == length - 1)
        {
            break;
        }
        index++;
    }
    close(file1);
    close(file2);
    return 0;
}
```

3.



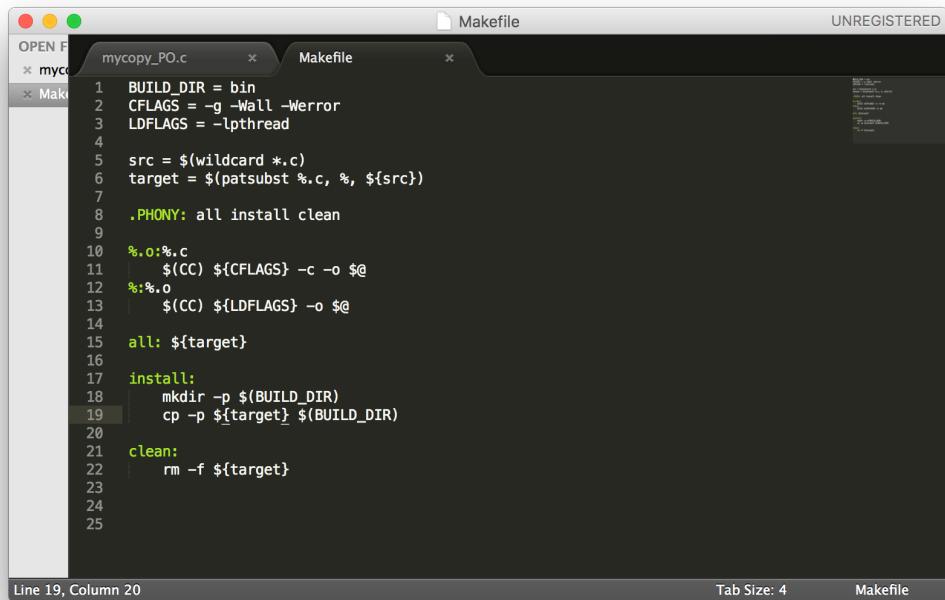
A screenshot of a Mac OS X terminal window titled "os_ex02 -- bash -- 80x24". The terminal shows the compilation of "mycopy_P0.c" into "mycopy_P0" using gcc, followed by running the program to copy "binary.png" to "copy.png". Finally, it lists the files in the current directory and shows the statistics for "binary.png" and "copy.png" using the "stat" command. The terminal output is as follows:

```
goudandeMBP:os_ex02 goudan$ make
gcc -c -o mycopy_P0 mycopy_P0.c
gcc -o mycopy_P0 mycopy_P0.o
goudandeMBP:os_ex02 goudan$ ./mycopy_P0 binary.png copy.png
goudandeMBP:os_ex02 goudan$ make
gcc -c -o mycopy_P0 mycopy_P0.c
gcc -o mycopy_P0 mycopy_P0.o
goudandeMBP:os_ex02 goudan$ ./mycopy_P0 binary.png copy.png
goudandeMBP:os_ex02 goudan$ ls
Makefile      copy.png      mycopy_P0      mycopy_P0.o
binary.png    mycopy.c     mycopy_P0.c    test01.txt
goudandeMBP:os_ex02 goudan$ stat copy.png
16777220 89817373 -rw-r--r-- 1 goudan staff 0 48762 "Oct 19 00:44:39 2017" "Sep
29 16:00:59 2017" "Oct 19 00:44:32 2017" "Sep 29 16:00:59 2017" 4096 96 0 copy.p
ng
goudandeMBP:os_ex02 goudan$ stat binary.png
16777220 88790458 -rw-r--r-- 1 goudan staff 0 48762 "Oct 19 00:44:36 2017" "Sep
29 16:00:59 2017" "Sep 29 16:01:00 2017" "Sep 29 16:00:59 2017" 4096 96 0 binary
.png
goudandeMBP:os_ex02 goudan$
```

```
os_ex02 -- bash -- 80x46
goudandeMBP:os_ex02 goudan$ diff mycopy.c mycopy_P0.c
1c1,7
< #include "stdio.h"
---
> #include <sys/stat.h>
> #include <sys/types.h>
> #include <stdlib.h>
> #include <stdio.h>
> #include <fcntl.h>
> #include <unistd.h>
>
4c10
<     FILE * file1,*file2;
---
>     int file1,file2;
6,7c12,13
<     file1 = fopen(argv[1],"rb");
<     file2 = fopen(argv[2],"wb");
---
>     file1 = open(argv[1],O_RDONLY);
>     file2 = open(argv[2],O_WRONLY);
15,17c21,23
<         fseek(file1,0,SEEK_END);
<         int length = ftell(file1);
<         //printf("%d\n",length);
---
>         lseek(file1,0,SEEK_END);
>         int length = lseek(file1, 0, SEEK_CUR);
>
20c26
<         while(!fseek(file1,index,SEEK_SET))
---
>         while(!lseek(file1,index,SEEK_SET))
22,23c28,29
<             fread(&c,1,1,file1);
<             fwrite(&c,1,1,file2);
---
>             read(file1,&c,1);
>             write(file2,&c,1);
30,31c36,37
<             fclose(file1);
<             fclose(file2);
---
>             close(file1);
>             close(file2);
goudandeMBP:os_ex02 goudan$
```

EX4

1.

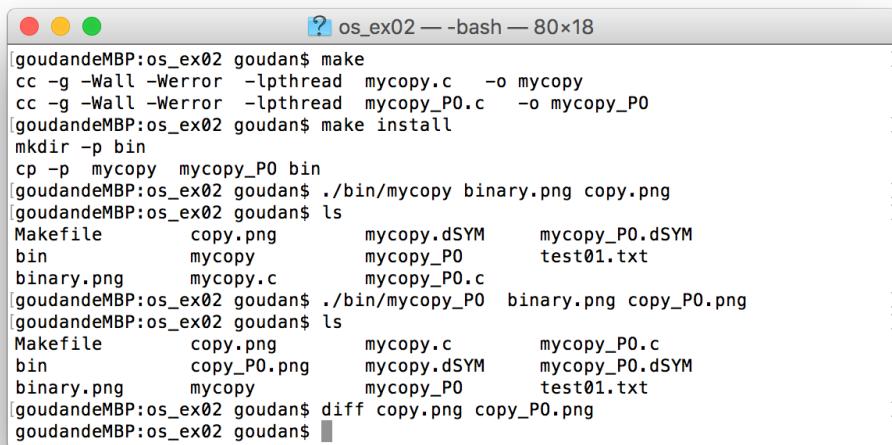


A screenshot of a Mac OS X desktop environment. In the foreground, a terminal window titled "os_ex02 — -bash — 80x18" is open. The command "ls" is being typed and is partially visible at the bottom of the screen. In the background, there is a dark-themed code editor window with tabs for "mycopy_PO.c" and "Makefile". The "Makefile" tab is active, showing the following content:

```
OPEN F... mycopy_PO.c Makefile UNREGISTERED
mycopy_PO.c
...
1 BUILD_DIR = bin
2 CFLAGS = -g -Wall -Werror
3 LDFLAGS = -lpthread
4
5 src = $(wildcard *.c)
6 target = $(patsubst %.c, ${BUILD_DIR}/%, ${src})
7
8 .PHONY: all install clean
9
10 %.o:%.c
11     $(CC) ${CFLAGS} -c -o $@
12 %:%.o
13     $(CC) ${LDFLAGS} -o $@
14
15 all: ${target}
16
17 install:
18     mkdir -p ${BUILD_DIR}
19     cp -p ${target} ${BUILD_DIR}
20
21 clean:
22     rm -f ${target}
23
24
25
```

The status bar at the bottom of the terminal window shows "Line 19, Column 20" and "Tab Size: 4".

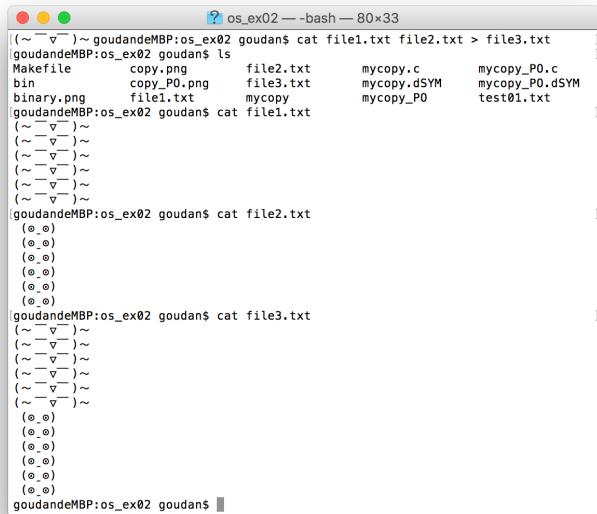
2.



A screenshot of a Mac OS X desktop environment. In the foreground, a terminal window titled "os_ex02 — -bash — 80x18" is open. The command "diff copy.png copy_P0.png" is being typed and is partially visible at the bottom of the screen. In the background, there is a dark-themed code editor window with tabs for "mycopy_PO.c" and "Makefile". The "Makefile" tab is active, showing the same content as in the previous screenshot. The status bar at the bottom of the terminal window shows "Line 19, Column 20" and "Tab Size: 4".

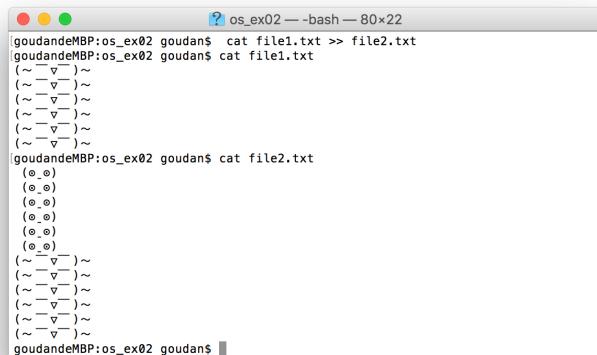
EX5

1.



```
[goudandeMBP:os_ex02 goudan$ cat file1.txt file2.txt > file3.txt]
[goudandeMBP:os_ex02 goudan$ ls
Makefile      copy.png       file2.txt      mycopy.c      mycopy_P0.c
bin          copy_P0.png    file3.txt      mycopy.dSYM   mycopy_P0.dSYM
binary.png   file1.txt     mycopy        mycopy_P0      test01.txt
[goudandeMBP:os_ex02 goudan$ cat file1.txt
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
[goudandeMBP:os_ex02 goudan$ cat file2.txt
(○.○)
(○.○)
(○.○)
(○.○)
(○.○)
(○.○)
[goudandeMBP:os_ex02 goudan$ cat file3.txt
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(○.○)
(○.○)
(○.○)
(○.○)
(○.○)
(goudandeMBP:os_ex02 goudan$ ]
```

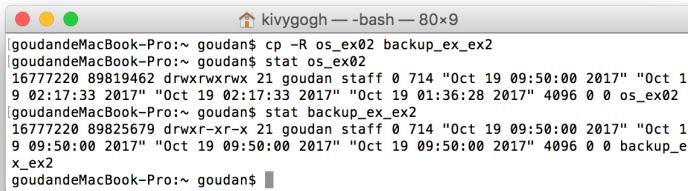
2.



```
[goudandeMBP:os_ex02 goudan$ cat file1.txt >> file2.txt
[goudandeMBP:os_ex02 goudan$ cat file1.txt
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
[goudandeMBP:os_ex02 goudan$ cat file2.txt
(○.○)
(○.○)
(○.○)
(○.○)
(○.○)
(○.○)
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(~_v_)~
(goudandeMBP:os_ex02 goudan$ ]
```

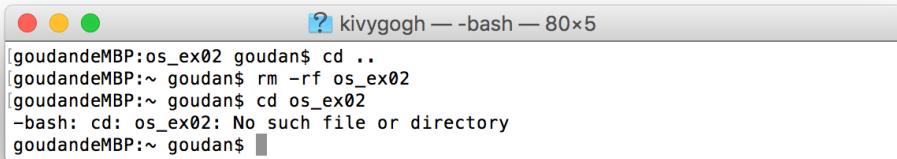
EX6

1.



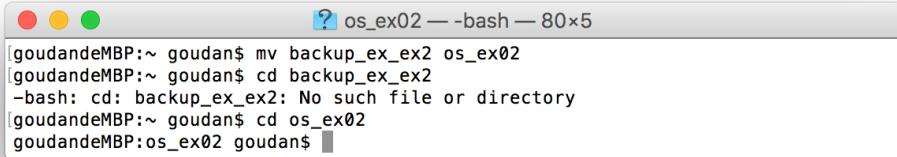
```
kivygogh — bash — 80x9
[goudandeMacBook-Pro:~ goudan$ cp -R os_ex02 backup_ex_ex2
[goudandeMacBook-Pro:~ goudan$ stat os_ex02
16777220 89819462 drwxrwxrwx 21 goudan staff 0 714 "Oct 19 09:50:00 2017" "Oct 1
9 02:17:33 2017" "Oct 19 02:17:33 2017" "Oct 19 01:36:28 2017" 4096 0 0 os_ex02
[goudandeMacBook-Pro:~ goudan$ stat backup_ex_ex2
16777220 89825679 drwxr-xr-x 21 goudan staff 0 714 "Oct 19 09:50:00 2017" "Oct 1
9 09:50:00 2017" "Oct 19 09:50:00 2017" "Oct 19 09:50:00 2017" 4096 0 0 backup_e
x_ex2
goudandeMacBook-Pro:~ goudan$ ]
```

1.



```
kivygogh — bash — 80x5
[goudandeMBP:os_ex02 goudan$ cd ..
[goudandeMBP:~ goudan$ rm -rf os_ex02
[goudandeMBP:~ goudan$ cd os_ex02
-bash: cd: os_ex02: No such file or directory
goudandeMBP:~ goudan$ ]
```

2.



```
os_ex02 — bash — 80x5
[goudandeMBP:~ goudan$ mv backup_ex_ex2 os_ex02
[goudandeMBP:~ goudan$ cd backup_ex_ex2
-bash: cd: backup_ex_ex2: No such file or directory
goudandeMBP:~ goudan$ cd os_ex02
goudandeMBP:os_ex02 goudan$ ]
```

EX7

1.

```
[goudandeMBP:os_ex02 goudan$ cd ..
[goudandeMBP:~ goudan$ ls -l
total 14184
drwxr-xr-x  4 goudan  staff   136  3  6  2017 Algorithm
drwxr-xr-x  6 goudan  staff   204  7 16 14:32 Applications
drwxr-xr-x  6 goudan  staff   204  4 23 2015 Calibre Library
drwxr-xr-x 11 goudan  staff   374 10 11 2016 Coffee-Slice
drwxr-xr-x 20 goudan  staff   680  9  2  2016 Design
drwx-----+ 79 goudan  staff  2686 10 19 01:42 Desktop
drwx-----+ 26 goudan  staff   884 10 18 23:45 Documents
drwx-----+ 186 goudan  staff  6324 10 19 01:06 Downloads
drwxr-xr-x  4 goudan  staff   136  7 17 18:04 GitHub
drwxr-xr-x 17 goudan  staff   578  7 17 19:14 Kinect
drwx-----@ 89 goudan  staff  3026 10 19 00:11 Library
drwxr-xr-x  9 goudan  staff   306  8 11 16:24 Motion-Detector-HTML5
drwxr-xr-x  8 goudan  staff   272  8  8  05:31 Motion-Detector-HTML5的副本
drwx-----+ 12 goudan  staff   408 12 14 2014 Movies
drwx-----+ 9 goudan  staff   306  9 13 11:07 Music
drwxr-xr-x 15 goudan  staff   510  7 17 18:55 OpenNI-MacOSX-x64-2.2
drwx-----+ 15 goudan  staff   510 12  3  2016 Pictures
drwxr-xr-x+ 5 goudan  staff   170  8 17 2014 Public
drwxr-xr-x  3 goudan  staff   102  8  8  19:37 PycharmProjects
drwxr-xr-x  3 goudan  staff   102  9 20 2015 Samsung
drwxr-xr-x 11 goudan  staff   374  2 24 2016 V3d
drwxr-xr-x 24 goudan  staff   816  8 22 12:54 anaconda
drwxr-xr-x  7 goudan  staff   238  9  6 12:32 angular2-quickstart
drwxr-xr-x  4 goudan  staff   136  1 11 2016 babel6
drwxr-xr-x  4 goudan  staff   136  3 28 2017 blog
drwxr-xr-x  2 goudan  staff    68  1 11 2016 bundle
drwxr-xr-x 13 goudan  staff   442  1 13 2016 buyshoes
drwxr-xr-x 15 goudan  staff   510  3  1  2016 buyshoes-react
drwxr-xr-x 49 goudan  staff  1666  9 15 07:01 chatBot
-rw-r--r--@ 1 goudan  staff  1998716  9 15 05:26 chatBot.zip
drwxr-xr-x  5 goudan  staff   170  8 12 05:28 csvtest
drwxr-xr-x 27 goudan  staff   918  1 27 2016 design2
drwxr-xr-x  5 goudan  staff   170  8 15 17:02 dp_num
drwxr-xr-x  5 goudan  staff   170  8 14 06:33 echarts_test
drwxr-xr-x  5 goudan  staff   170  3 11 2017 eclipse
drwxr-xr-x  7 goudan  staff   238 10 14 17:05 ex1
-rw-r--r--@ 1 goudan  staff  46269 10  9 13:02 ex1.pdf
drwxr-xr-x  4 goudan  staff   136  3  4  2017 game
```

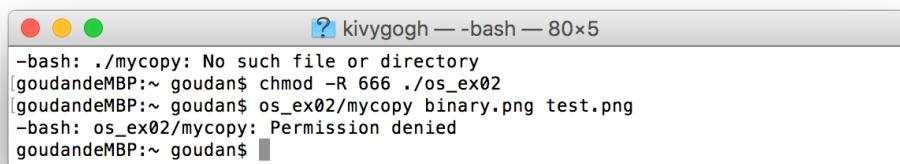
2.

```
[goudandeMBP:~ goudan$ cd os_ex02
[goudandeMBP:os_ex02 goudan$ umask
0022
[goudandeMBP:os_ex02 goudan$ umask -S
u=rwx,g=rx,o=rx
[goudandeMBP:os_ex02 goudan$
[goudandeMBP:os_ex02 goudan$ umask 002
[goudandeMBP:os_ex02 goudan$ umask
0002
[goudandeMBP:os_ex02 goudan$ ]
```

The umask acts as a set of permissions that applications cannot set on files.
It's a file mode creation mask for processes and cannot be set for directories

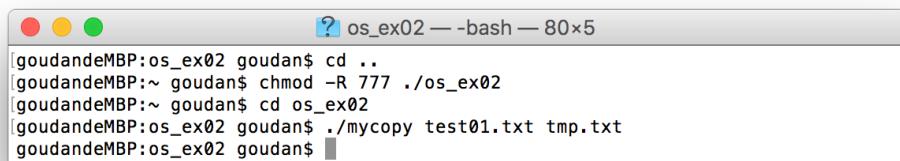
itself. Most applications would not create files with execute permissions set, so they would have a default of 666, which is then modified by the umask.

3.



```
? kivygogh — -bash — 80x5
[bash: ./mycopy: No such file or directory
[goudandeMBP:~ goudan$ chmod -R 666 ./os_ex02
[goudandeMBP:~ goudan$ os_ex02/mycopy binary.png test.png
[bash: os_ex02/mycopy: Permission denied
[goudandeMBP:~ goudan$ ]]
```

4.



```
? os_ex02 — -bash — 80x5
[goudandeMBP:os_ex02 goudan$ cd ..
[goudandeMBP:~ goudan$ chmod -R 777 ./os_ex02
[goudandeMBP:~ goudan$ cd os_ex02
[goudandeMBP:os_ex02 goudan$ ./mycopy test01.txt tmp.txt
[goudandeMBP:os_ex02 goudan$ ]]
```