

**Table of Content**

[Q1. Type of file 2](#_Toc168529989)

[Q2. File\_access.c 4](#_Toc168529990)

[Q3. Umask 5](#_Toc168529991)

[Q4. Open and unlink 6](#_Toc168529992)

[Q5. Exercise 4.2 7](#_Toc168529993)

[Q6. Exercise 4.3 8](#_Toc168529994)

[Q7. Exercise 4.4 8](#_Toc168529995)

[Q8. Exercise 4.8 8](#_Toc168529996)

[Q9. Exercise 4.9 9](#_Toc168529997)

Color and symbol specification in Terminal output of this document:

1. gcc hw1\_h.c No shading color: command line;
2. Hello World Yellow shading color: program output;
3. d Blue highlight with yellow shading: (waiting) user input;
4. 🡸 Redirect the STDOUT Red shading: author extra explanation warning
5. $ --- command prompt for NON-root account;
6. # --- command prompt for Root account.

# Q1. Type of file

|  |
| --- |
| ***hw1\_type\_of file.c***  The program is structured to cycle through a list of filenames given as command-line arguments, and it outputs the type of each file. It employs the `lstat` system call to fetch the status details for every file, storing this data in a `struct stat` buffer. The program then determines the file type, assigns a matching string description to a pointer, and prints the corresponding type in the terminal. |
| #include"apue.h"  int main (int argc,char \*\*argv)  {  int i;  struct stat buf; // a *“struct stat”* type buffer to store the file’s information  char \*ptr;  // Loop through the command-line arguments, starting from the first file name argument  for(i=1;i<argc;i++)  {  printf("\n%s: ",argv[i]); // print the file name  // get file status information through *lstat()* and store it in buf  if(lstat(argv[i],&buf)<0)  {  err\_ret("lstat error");  continue;  }  // check the file type and set the corresponding description string  if(S\_ISREG(buf.st\_mode))  ptr="regular file";  else if(S\_ISDIR(buf.st\_mode))  ptr="directory file";  else if(S\_ISCHR(buf.st\_mode))  ptr="character special";  else if(S\_ISBLK(buf.st\_mode))  ptr="block special";  else if(S\_ISFIFO(buf.st\_mode))  ptr="fifo ";  else if(S\_ISLNK(buf.st\_mode))  ptr="symbolic link";  else if(S\_ISSOCK(buf.st\_mode))  ptr="socket";  else  ptr="\*\* unknown mode \*\*";  // print out the file type  printf("%s\n",ptr);  }  exit(0);  } |

**[Terminal Coding Tests]**

|  |  |
| --- | --- |
| $  $  > | gcc hw1\_type\_of file.c -o hw1  ./hw1 /etc/passwd /etc /dev/log /dev/tty \  /var/lib/oprofile/opd\_pipe /dev/sr0 /dev/cdrom  /etc/passwd: regular file  /etc: directory file  /dev/log: symbolic link  /dev/tty: character special  /var/lib/oprofile/opd\_pipe: lstat error: No such file or directory  /dev/sr0: block special  /dev/cdrom: symbolic link |

**[Output analysis]**

We can see the output is little different from the text book. It may be resulted by the different system which is working with the program. Since I don’t have the *opd\_pipe* file, so a warning is given in my terminal. Apart from that, other files’ type is correctly printed.

# Q2. File\_access.c

|  |
| --- |
| ***Hw2\_file\_access.c***  This program checks the read access permissions and the ability to open a specified file for reading. It takes a single file path as a command-line argument and first uses the access function to check if the file can be read, printing an error message if it cannot. Then, it attempts to open the file with read-only permissions using the open function, printing an error message if this fails. If both checks succeed, it confirms that read access and opening for reading are both okay. |
| #include "apue.h"  #include <fcntl.h>  int  main(int argc, char \*argv[])  {  // if the number of arguments in cmd is not equal to 2, print a error message  if (argc != 2)  printf("usage: a.out <pathname>");  // if the user has the read permission to the file in the argv[1], then print "read access OK ", otherwise print "access error for this file"  if (access(argv[1], R\_OK) < 0)  printf("access error for %s", argv[1]);  else  printf("read access OK\n");  // if the user has the readonly permission to the file in the argv[1], then print " open for reading OK ", otherwise print " open error for this file"  if (open(argv[1], O\_RDONLY) < 0)  printf("open error for %s", argv[1]);  else  printf("open for reading OK\n");  exit(0);  } |

**[Terminal Coding Tests]**

|  |  |
| --- | --- |
| $  $  $  $  # | gcc hw2\_file\_access.c -o hw2  ./hw2 hw2\_file\_access.c  read access OK  open for reading OK    ./hw2 /etc/shadow  access error for /etc/shadow  open error for /etc/shadow    su root  Password:    ./hw2 /etc/shadow  read access OK  open for reading OK |

**[Output analysis]**

I use this program to check the user account’s permission to the *hw2\_file\_access.c* file, and it indicates that the user account has the read and open permission. However, when I check the permission the the */etc/shadow* file, it indicated errors which means the user account dosen’t has the permission to */etc/shadow* file.

Then I changed to the root account for testing the program, finding that the root account has the read and open access to */etc/shadow* file.

# Q3. Umask

|  |
| --- |
| ***Hw3\_umask.c***  This program demonstrates the use of the umask function to control file creation permissions. It first sets the umask to 0 and creates a file named "foo" with read and write permissions for the user, group, and others. Then, it sets a new umask to remove read and write permissions for the group and others and creates another file named "bar" with the same initial permissions, which will actually result in permissions of due to the new umask. |
| #include "apue.h"  #include <fcntl.h>  #define RWRWRW (S\_IRUSR|S\_IWUSR|S\_IRGRP|S\_IWGRP|S\_IROTH|S\_IWOTH)  int  main(void)  {  // set umask to 0  umask(0);  // create foo with 0666 permision  if (creat("foo", RWRWRW) < 0)  err\_sys("creat error for foo");  // reset umask to 022  umask(S\_IRGRP | S\_IWGRP | S\_IROTH | S\_IWOTH);  // create foo with 0666 permision  if (creat("bar", RWRWRW) < 0)  err\_sys("creat error for bar");  } |

**[Terminal Coding Tests]**

|  |  |
| --- | --- |
| $  $  $ | gcc hw3.c -o hw3  ./hw3  ls -l foo bar  -rw------- 1 root root 0 Jun 5 23:27 bar  -rw-rw-rw- 1 root root 0 Jun 5 23:27 foo |

**[Output analysis]**

We can find that foo is with RWRWRW permission, while bar only has RW permission for the users and no read and write permission for group and other users. That is because:

1. When creating foo, the actual permission is RWRWRW-UMASK, which is 0666 – 0 = 0666.
2. When creating bar, the actual permission is RWRWRW-UMASK, which is RWRWRW – umask(S\_IRGRP | S\_IWGRP | S\_IROTH | S\_IWOTH) meaning when files are created, the read and write permissions for the group and others will be masked out.

# Q4. Open and unlink

|  |
| --- |
| ***Hw4\_open\_and\_unlink.c***  This program opens a file named "tempfile" for reading and writing, then immediately unlinks (deletes) the file. Despite the file being unlinked, the file descriptor remains open, allowing the program to continue accessing the file's contents. The program prints a message indicating the file has been unlinked, then sleeps for 15 seconds before printing done and exiting. |
| #include "apue.h"  #include <fcntl.h>  int  main(void)  {  // open the tempfile  if (open("tempfile", O\_RDWR) < 0)  err\_sys("open error");  // unlink the tempfile  if (unlink("tempfile") < 0)  err\_sys("unlink error");  printf("file unlinked\n");  sleep(15);  printf("done\n");  exit(0);  } |

**[Terminal Coding Tests]**

|  |  |  |  |
| --- | --- | --- | --- |
| $  $ | gcc hw4.c -o hw4  hw4  file unlinked  done | $    $  $ | df /home  Filesystem 1K-blocks Used Available Use% Mounted on  /dev/sda3 61091660 11972920 45983048 21% /  ls -l tempfile  -rw-r--r-- 1 root root 108 Jun 5 23:44 tempfile  df /home  Filesystem 1K-blocks Used Available Use% Mounted on  /dev/sda3 61091660 11972916 45983052 21% / |

**[Output analysis]**

Before we running the program, we can find that the used memory is 11972920 bytes, however after we unlink the *tempfile* file, the used memory reduced to 11972916 bytes. And the available memory increased from 45983048 bytes to 45983052 bytes.

# Q5. Exercise 4.2

What happens if the file mode creation mask is set to 777 (octal)? Verify the results using your shell’s umask command.

|  |  |
| --- | --- |
| $  $  $  $ | umask  0002  umask 777  echo "Hello SFBU" > hw5.txt  ls -l hw5.txt  ---------- 1 beza beza 11 Jun 4 00:03 hw5.txt  vim hw5.txt  Permission Denied |

After we set the umask to 777, the permission bits for the new created files will be come all ----------, which means that all permissions for the user, group and other users are all mask out. Even though we wrote some thing into the newly created files, once done and created, we are no longer have the access to this file and cannot see the content.

# Q6. Exercise 4.3

Verify that turning off user-read permission for a file that you own denies your access to the file.

|  |  |
| --- | --- |
| $  $  $  $  $ | echo "Hello SFBU hw666" > hw6.txt  ls -l hw6.txt  -rw-rw-r-- 1 beza beza 17 Jun 4 00:13 hw6.txt  cat hw6.txt  Hello SFBU hw666  chmod u-r hw6.txt 🡨 turning off user-read permission  cat hw6.txt  cat: hw6.txt: Permission denied |

# Q7. Exercise 4.4

When the program is executed, if the files "foo" and "bar" already exist, the creat() function behaves similarly to the open(path, O\_WRONLY | O\_CREAT | O\_TRUNC, mode) function call.

Before running the program again, if we input some characters into "foo" and "bar", these files will still persist after the program is executed. However, the content in "foo" will be truncated. This is due to the creat() function used in the program to generate the "foo" and "bar" files. If these files already exist, creat() truncates them to zero length. As a result, any data previously written to "foo" will be erased when the program is run.

# Q8. Exercise 4.8

|  |  |
| --- | --- |
| $  $  $ | du -sh /home/beza/CS510-APUE/Homework/week04  592K /home/beza/CS510-APUE/Homework/week04  du -sh /home/beza/CS510-APUE/Homework/week04  588K /home/beza/CS510-APUE/Homework/week04<- When running the program  du -sh /home/beza/CS510-APUE/Homework/week04  588K /home/beza/CS510-APUE/Homework/week04 <- After running the program |
| $  $  $ | du -sh /home/beza/CS510-APUE/Homework/week04  592K /home/beza/CS510-APUE/Homework/week04  df /home/beza/CS510-APUE/Homework/week04  Filesystem 1K-blocks Used Available Use% Mounted on  /dev/sda3 61091660 11973320 45982648 21% / <- When running the program  df /home/beza/CS510-APUE/Homework/week04  Filesystem 1K-blocks Used Available Use% Mounted on  /dev/sda3 61091660 11973316 45982652 21% / <- After running the program |

The df /home command reads the metadata of the disk partition that contains the specified folder, rather than the actual directory. On the other hand, the du command traverses the specified directory tree and calculates the total size of all files within the directory, thereby returning the total space that the directory occupies.

Although both commands can determine the size of data files, du depends on the directory file to locate these data files, while df does not.

Once the program unlinks “tempfile”, the file is removed. Consequently, if we use du to check the size of all files in the current directory, “tempfile” will not be included. This means that the outputs of the du -sh command executed before (which is indicated by 588K when running) and after “done” (which is indicated by 588K after running) is printed on the console will be identical. However, for df function, the metadata indicated different numbers for the space and useages.

# Q9. Exercise 4.9

The *unlink()* operation modifies the link count of a file, which is a piece of information stored in the file's inode. This change also results in an update to the *st\_ctim* attribute of the file. The *unlink()* system call is designed to remove a link, which is essentially a mapping from a name to a file. If a file only has one link, it gets deleted when *unlink()* is called. However, if there are multiple links, the *unlink()* operation only removes the directory entry of the filename, not the file itself. In such cases, the *unlink()* operation updates the changed-status time of the file associated with the removed name.

|  |  |
| --- | --- |
| $  $  $  $  $ | touch hw9  ln hw9 ex4\_9  stat hw9  File: hw9  Size: 0 Blocks: 0 IO Block: 4096 regular empty file  Device: 803h/2051d Inode: 3146647 Links: 2  Access: (0664/-rw-rw-r--) Uid: ( 1000/ beza) Gid: ( 1000/ beza)  Access: 2024-06-04 01:07:07.695974868 -0700  Modify: 2024-06-04 01:07:07.695974868 -0700  Change: 2024-06-04 01:07:29.632102144 -0700  Birth: 2024-06-04 01:07:07.695974868 -0700  rm ex4\_9  stat hw9  File: hw9  Size: 0 Blocks: 0 IO Block: 4096 regular empty file  Device: 803h/2051d Inode: 3146647 Links: 1  Access: (0664/-rw-rw-r--) Uid: ( 1000/ beza) Gid: ( 1000/ beza)  Access: 2024-06-04 01:07:07.695974868 -0700  Modify: 2024-06-04 01:07:07.695974868 -0700  Change: 2024-06-04 01:08:07.920322529 -0700  Birth: 2024-06-04 01:07:07.695974868 -0700 |

[[19.02 18.97 92.38 34.59 46.18 0.61 71. 57.31 45.81 5.73] [92.3 70.65 88.28 53.08 9.71 87.39 32.24 97.99 59.49 47.69] [51.08 71.22 75.33 6.42 22. 75.63 39.96 15.09 58.1 86.44]

[ 7.03 63.22 82.58 90.17 82.13 88.53 41.61 46.7 13.35 31.51] [97.31 48.04 60.07 67.96 88.65 72.38 43.77 32.03 86.24 28.27] [33.09 50.14 7.92 63.92 82.83 7.11 25.07 89.76 62.35 22.05]

[38.48 37.93 9.56 4.1 63.33 20.86 66.62 2.6 68.67 13.1 ] [89.71 49.59 14.68 26.45 73.58 40.46 2.66 49.2 92.08 25.3 ] [87.09 13.38 64.88 45.96 50.52 15.08 47.24 98.03 59.54 80.58]

[52.77 85.03 25.84 73.8 44.5 32.21 27.31 67.73 30.69 53.24]] [[12. 31.85 3.92 3.22 84.12 1.24 4.96 23.36 48.52 72.75] [25.65 95.97 3.55 60.61 35.86 45.62 34.39 26.82 69.24 30.55]

[94.96 95.62 0.33 30.61 72.92 23.72 95.84 25.48 11.04 81.95] [77.75 5.71 25.78 18.4 78.61 77.69 14.19 65.28 18.1 71.3 ] [ 2.11 90.83 70.97 51.39 47.65 21.57 4.52 44.39 90.22 77.97]

[25.3 73.23 85.52 15.85 29.62 10.89 29.98 8.36 7.68 88.42] [51.73 65.56 71.81 37.48 45.01 85.82 70.15 58.31 65.14 25.48] [88.2 64.59 30.42 30.82 95.68 51.3 65.79 25.7 96.93 47.21]

[71.62 36.79 91.55 82.75 1.51 44.01 94.29 11.82 3.83 66.85] [ 5.79 4.32 91.9 21.11 88.88 17.4 18.93 90.38 98.58 5.68]] [[60.86 87.76 59.07 95.83 31.33 31.39 11.56 6.37 52.14 27.56]

[13.62 20.1 68.95 94.72 11.85 59.87 2.71 10.43 60.36 68.7 ] [43.18 21.09 3.65 76.65 95.01 70.08 77.23 23.07 81.27 11.1 ] [ 4.2 12.8 64.03 6.34 41.58 57.68 21.16 13.78 53.57 5.71]

[43.19 20.31 25.74 96.38 20.18 2.86 94.63 76.08 8.77 51.32] [38.07 20.36 15.47 53.56 38.09 11.83 5.51 29.73 78.31 96. ] [75.02 71.65 42.51 66.64 38.14 25.21 68.36 48.99 8.62 46.66]

[71.38 58.8 41.04 96.42 45.62 64.94 25.88 20.45 52.95 16.95] [85.43 11.46 63.93 78.4 73.22 59.67 90.89 89.05 95.13 51.45] [52.66 47.98 44.58 75.48 65.81 79.61 40.71 48.89 54.91 97.98]] [[ 1.68 24.23 75.25 86.55 9.64 7.8 0.69 30.88 87.21 79.74] [34.19 58.9 78.74 28.28 19.61 40.81 67.49 40.16 48.3 27.79] [83.67 17. 51.88 59.64 58.35 74.78 81.57 18.58 70.05 58.81]

[20.46 90.51 9.7 2.23 19.88 63.63 80.86 6.86 14.61 50.72] [52.84 44.67 26.99 41.1 68.68 11.7 35.92 62.72 78.89 70.95] [46.77 79.81 18.6 75.88 18.61 14.05 98.12 88.74 96.98 83.72]

[87.05 76.21 76.21 96.51 48.47 36.97 92.46 68.7 89.97 5.23] [80.04 49.29 33.73 36.42 25.51 4.34 14.58 23.57 62.23 9.82] [ 9.57 27.62 96.67 38.56 14.27 8.57 35.82 65.03 50.69 34.96]

[28.98 45.44 41.75 28.53 50.6 22.67 90.85 53.68 55.47 48.08]] [[82.06 83.61 53.71 40.64 34.41 32.96 44.18 80.65 63.57 90.91] [65.99 12.11 88.56 24.75 62.51 21.72 37.26 54.69 3.71 57.91]

[29.99 47.24 54.77 72.91 16.39 98.11 47.64 47.07 56.23 87.98] [97.25 9.49 18.34 72.32 81.01 60.87 97.39 77.12 15.21 5.82] [ 2.18 16.19 2.48 33.63 61.12 37.32 21.62 5.25 94.42 48.97]

[85.68 81.56 95.04 96.36 37.52 15.14 9.21 44.66 4.8 53.69] [35.93 1.82 79.99 94.58 53.45 0.68 4.91 85.49 7.52 18.66] [15.73 11.19 78.51 35.16 78.35 6.62 63.34 6.9 41.1 24.8 ] [90.36 29.16 7.24 77.89 6.09 76.33 75.65 31.38 16.64 30.83] [ 9.83 73.21 11.6 68.56 91.38 56.11 27.76 44.67 76.29 26.2 ]] [[86.2 46.02 97.18 6.9 53.82 54.97 9.9 26.6 85.47 58.29] [33.1 17.55 77.59 22.36 61.3 13.58 13.44 3.33 20.21 25.45] [63.29 79.45 7.35 51.87 61.89 14.43 67.64 6.65 16.72 94.32] [89.56 33.85 75.74 63.36 33.66 75.04 29.86 8.77 37.19 43.83]

[85.58 65.96 72.05 19.15 29.62 35.09 7.36 61.91 91.5 34.35] [42.38 47.97 13.97 23.13 74.18 13.92 91.15 65.59 3.49 40.03] [84.14 34.84 56.65 72.4 1.87 97.06 77.14 12.74 76.69 8.68]

[85.43 83.32 30.79 10.45 95.92 5.35 16.17 80.36 93.29 59.15] [ 8.64 31.02 91.47 19.19 55.45 95.08 62.39 51.99 7.3 30.25] [ 3.7 76.95 59.58 98.93 36.73 97.01 73.59 12. 19.26 4.78]] [[19.02 18.97 92.38 34.59 46.18 0.61 71. 57.31 45.81 5.73] [92.3 70.65 88.28 53.08 9.71 87.39 32.24 97.99 59.49 47.69] [51.08 71.22 75.33 6.42 22. 75.63 39.96 15.09 58.1 86.44]

[ 7.03 63.22 82.58 90.17 82.13 88.53 41.61 46.7 13.35 31.51] [97.31 48.04 60.07 67.96 88.65 72.38 43.77 32.03 86.24 28.27] [33.09 50.14 7.92 63.92 82.83 7.11 25.07 89.76 62.35 22.05]

[38.48 37.93 9.56 4.1 63.33 20.86 66.62 2.6 68.67 13.1 ] [89.71 49.59 14.68 26.45 73.58 40.46 2.66 49.2 92.08 25.3 ] [87.09 13.38 64.88 45.96 50.52 15.08 47.24 98.03 59.54 80.58]

[52.77 85.03 25.84 73.8 44.5 32.21 27.31 67.73 30.69 53.24]] [[12. 31.85 3.92 3.22 84.12 1.24 4.96 23.36 48.52 72.75] [25.65 95.97 3.55 60.61 35.86 45.62 34.39 26.82 69.24 30.55]

[94.96 95.62 0.33 30.61 72.92 23.72 95.84 25.48 11.04 81.95] [77.75 5.71 25.78 18.4 78.61 77.69 14.19 65.28 18.1 71.3 ] [ 2.11 90.83 70.97 51.39 47.65 21.57 4.52 44.39 90.22 77.97]

[25.3 73.23 85.52 15.85 29.62 10.89 29.98 8.36 7.68 88.42] [51.73 65.56 71.81 37.48 45.01 85.82 70.15 58.31 65.14 25.48] [88.2 64.59 30.42 30.82 95.68 51.3 65.79 25.7 96.93 47.21]

[71.62 36.79 91.55 82.75 1.51 44.01 94.29 11.82 3.83 66.85] [ 5.79 4.32 91.9 21.11 88.88 17.4 18.93 90.38 98.58 5.68]] [[60.86 87.76 59.07 95.83 31.33 31.39 11.56 6.37 52.14 27.56]

[13.62 20.1 68.95 94.72 11.85 59.87 2.71 10.43 60.36 68.7 ] [43.18 21.09 3.65 76.65 95.01 70.08 77.23 23.07 81.27 11.1 ] [ 4.2 12.8 64.03 6.34 41.58 57.68 21.16 13.78 53.57 5.71]

[43.19 20.31 25.74 96.38 20.18 2.86 94.63 76.08 8.77 51.32] [38.07 20.36 15.47 53.56 38.09 11.83 5.51 29.73 78.31 96. ] [75.02 71.65 42.51 66.64 38.14 25.21 68.36 48.99 8.62 46.66]

[71.38 58.8 41.04 96.42 45.62 64.94 25.88 20.45 52.95 16.95] [85.43 11.46 63.93 78.4 73.22 59.67 90.89 89.05 95.13 51.45] [52.66 47.98 44.58 75.48 65.81 79.61 40.71 48.89 54.91 97.98]] [[ 1.68 24.23 75.25 86.55 9.64 7.8 0.69 30.88 87.21 79.74] [34.19 58.9 78.74 28.28 19.61 40.81 67.49 40.16 48.3 27.79] [83.67 17. 51.88 59.64 58.35 74.78 81.57 18.58 70.05 58.81]

[20.46 90.51 9.7 2.23 19.88 63.63 80.86 6.86 14.61 50.72] [52.84 44.67 26.99 41.1 68.68 11.7 35.92 62.72 78.89 70.95] [46.77 79.81 18.6 75.88 18.61 14.05 98.12 88.74 96.98 83.72]

[87.05 76.21 76.21 96.51 48.47 36.97 92.46 68.7 89.97 5.23] [80.04 49.29 33.73 36.42 25.51 4.34 14.58 23.57 62.23 9.82] [ 9.57 27.62 96.67 38.56 14.27 8.57 35.82 65.03 50.69 34.96]

[28.98 45.44 41.75 28.53 50.6 22.67 90.85 53.68 55.47 48.08]] [[82.06 83.61 53.71 40.64 34.41 32.96 44.18 80.65 63.57 90.91] [65.99 12.11 88.56 24.75 62.51 21.72 37.26 54.69 3.71 57.91]

[29.99 47.24 54.77 72.91 16.39 98.11 47.64 47.07 56.23 87.98] [97.25 9.49 18.34 72.32 81.01 60.87 97.39 77.12 15.21 5.82] [ 2.18 16.19 2.48 33.63 61.12 37.32 21.62 5.25 94.42 48.97]

[85.68 81.56 95.04 96.36 37.52 15.14 9.21 44.66 4.8 53.69] [35.93 1.82 79.99 94.58 53.45 0.68 4.91 85.49 7.52 18.66] [15.73 11.19 78.51 35.16 78.35 6.62 63.34 6.9 41.1 24.8 ] [90.36 29.16 7.24 77.89 6.09 76.33 75.65 31.38 16.64 30.83] [ 9.83 73.21 11.6 68.56 91.38 56.11 27.76 44.67 76.29 26.2 ]] [[86.2 46.02 97.18 6.9 53.82 54.97 9.9 26.6 85.47 58.29] [33.1 17.55 77.59 22.36 61.3 13.58 13.44 3.33 20.21 25.45] [63.29 79.45 7.35 51.87 61.89 14.43 67.64 6.65 16.72 94.32] [89.56 33.85 75.74 63.36 33.66 75.04 29.86 8.77 37.19 43.83]

[85.58 65.96 72.05 19.15 29.62 35.09 7.36 61.91 91.5 34.35] [42.38 47.97 13.97 23.13 74.18 13.92 91.15 65.59 3.49 40.03] [84.14 34.84 56.65 72.4 1.87 97.06 77.14 12.74 76.69 8.68]

[85.43 83.32 30.79 10.45 95.92 5.35 16.17 80.36 93.29 59.15] [ 8.64 31.02 91.47 19.19 55.45 95.08 62.39 51.99 7.3 30.25] [ 3.7 76.95 59.58 98.93 36.73 97.01 73.59 12. 19.26 4.78]]