Test cases

Default Work queue length = 10
Default trashbin size = 10
Path of trashbin folder: /.trashbin/

Test case 1
Command: ./execve sync.sh
Description : 5 files < 4096 KB (named syncinf1.txt …. syncinf5.txt) with different combinations of clone flags are removed to showcase the synchronous behaviour. There will be no use of work queues. Files will be moved directly to trashbin according to flags synchronously.
Output : Creates 5 files in /.trashbin/ with appropriate naming convention.

Test case 2
Command: ./execve async_with_work_queue.sh
Description: 10 files (named inf1.txt … inf10.txt) > 4096 KB with different combination of flags are removed to showcase the asynchronous behaviour using work queue.
Output: Creates 10 files in /.trashbin/ with appropriate naming convention.

Test case 3
Command: ./execve async_with_work_wait_queue.sh
cat /proc/workqueue_info can be used to check current processes in the work queue
Description: 17 files (named inf1.txt … inf17.txt) > 4096 KB with different combination of flags are removed to showcase the asynchronous behaviour using work queue and wait queue. The processes will be moved to wait queue if work queue is maxed out.
Output: Creates 10 files in /.trashbin/ with appropriate naming convention.

Test case 4
Command: ./execve sync_async.sh
Description: 5 files with different sizes (> 4096 KB and < 4096 KB) with different combination of flags are removed to showcase the asynchronous synchronous behaviour using work queue.
Output: Creates 5 files in /.trashbin/ with appropriate naming convention.

Test case 5
Command: ./execve empty.sh
Description: 3 empty files (named empty1.txt … empty3.txt) (size = 0) with different combination of flags are removed. Code works even with empty files.
Output: Creates 3 files in /.trashbin/ with appropriate naming convention.

Test case 6 : ioctl is fired when all the operations on the file are completed
Command: ./ioctl -u filename
Description: ioctl command for undeleting a file is fired. It will delete the file from trashbin folder and move it to its original location with original name. We can fire the ioctl for the file which is present in other than hw3 to check the validity of restoration of file to its original position.
Output: File will be restored to original location.

Test case 7 : ioctl is fired when the file has been removed from the queue and is in middle of processing
Command: ./ioctl -u filename
Description: ioctl command for undeleting a file is fired. We will allow the file to complete the processing and then It will delete the file from trashbin folder and move it to its original location with original name.
Output: File will be restored to original location.

Test case 8 : ioctl is fired when the file is queued
Command: ./ioctl -u filename
Description: ioctl command for undeleting a file is fired. As the file is still queued , no processing has started on the file and user wants to restore it. Efficient way is don't process the file and remove the file from the queue.
Output: File was never moved from its original position.

Test case 9
Command: echo "3" > /proc/trashb_max_size.c
Description: It will reduce the size of trashbin folder from 10 to 3. In this case older files according to timestamp (oldest) will be deleted from trashbin by the cleaner thread
Output: Only 3 latest files will remain in trashbin folder.

Test case 10
Command: echo "3" > /proc/trashb_max_qlen.c
Description: It will reduce the size of work queue from 10 to 3. In this case, items already in queue will be processed normally irrespective of size. New items will go into wait queue if work queue will be full.
Output: List content of queue (using cat /proc/workqueue_info) to check size.

Test case 11
Command: ioctl
Description: It will delete all the files, belonging to that user, from trashbin folder. User will not be allowed to delete other user's files. Only root can purge trashbin folder completely.
Output: Files will be deleted from trashbin for that user.

Test case 12

Command: ls , stat, cat, ll -a, find

Description: User can list, stat, cat only those files which belongs to it. It doesn't have permission to perform operations on other user's file. Root has permission to perform operations on all files.