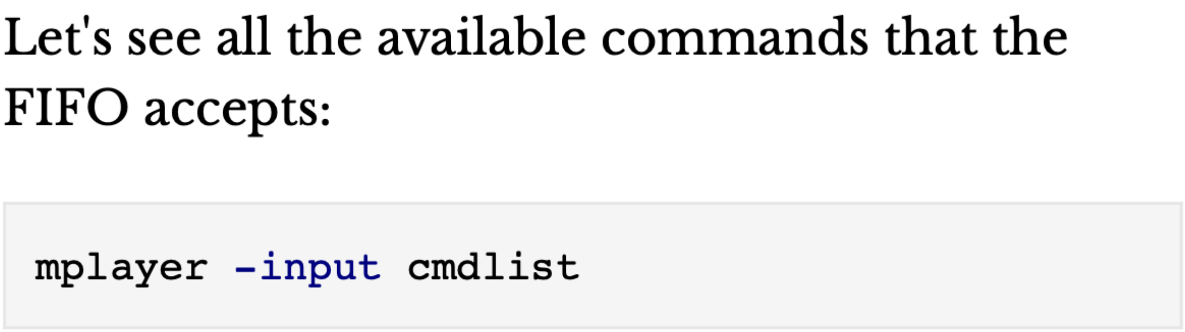
ECE 5725 Lab1 Week2 Report (Homework 2)

Names: Zixiao Wang, Yue Zhang
NetID: zw579, yz2455

Date: September 28, 2019

1. First, FIFO is named-PIPE that exists in the filesystem just like a regular file while PIPE doesn’t exist in the filesystem.
   Second, FIFO will keep existing in filesystem when process ends but PIPE won’t last.
   Third, PIPE is directional but FIFO can be unidirectional and bidirectional.
   Finally, PIPE is only used for communication between related processes with common ancestors while FIFO can be used for any unrelated processes, even across different network or computer, which is what we cannot do with PIPE.

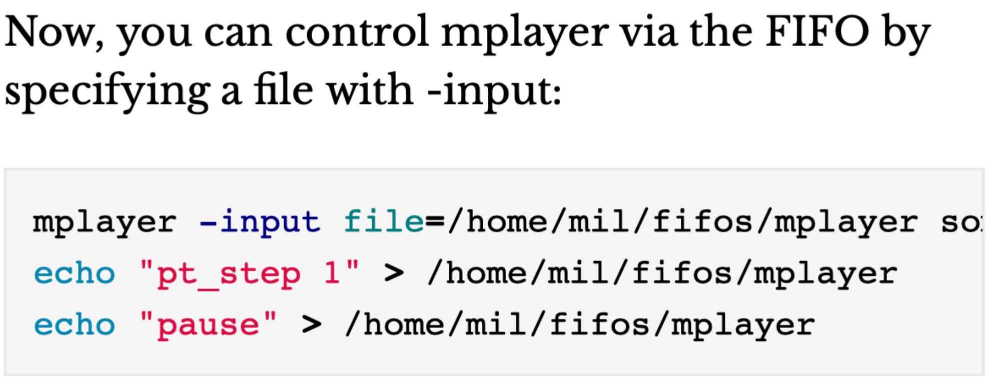
2.



Let's see all the available commands that the FIFO accepts:

```
mplayer -input cmdlist
```

The first confusing part is the guide said this command will give you all available commands that **FIFO** accepts, however, it actually lists all commands that **mplayer** accepts!



Now, you can control mplayer via the FIFO by specifying a file with -input:

```
mplayer -input file=/home/mil/fifos/mplayer so
echo "pt_step 1" > /home/mil/fifos/mplayer
echo "pause" > /home/mil/fifos/mplayer
```

The second confusing part is the guide suggests all three command lines should be typed in the same console, but if you do that it won’t work. Because we have to type the first line in one console and the last two in another. The first line tells mplayer to read command from fifo file, the last two lines send commands to fifo file.

3. Touch screen:
   Pin#3 SDA: Data Line of I2C
   Pin#5 SCL: Clock Line of I2C
   Pin#18 RT_INT

   Buttons:
   Pin#15: GPIO22
   Pin#13: GPIO27
   Pin#11: GPIO17
   Pin#16: GPIO23

   Serial Logic:
   Pin#21 MISO: Master Input Slave Output of SPI
   Pin#19 MOSI: Master Output Slave Input of SPI
   Pin#22 TFT_DC_3V: TFT 3.3V Power Supply
   Pin#24 TFT_CS_3V: TFT Chip Select
   Pin#23 SCLK: Serial Clock

   Touch screen control:
   Pin#18 RT_INT, reset
   Pin#12: GPIO18, control a diode
   Pin#26: RT_CS_3V, chip select

4. All possible GPIO pins may be used(maximum set):
   pin#3(GPIO2), pin#5(GPIO3), pin#7(GPIO4), pin#29(GPIO5), pin#31(GPIO6), pin#26(GPIO7), pin#24(GPIO8), pin#21(GPIO9), pin#19(GPIO10), pin#23(GPIO11), pin#32(GPIO12), pin#33(GPIO13), pin#8(GPIO14), pin#10(GPIO15), pin#36(GPIO16), pin#11(GPIO17), pin#12(GPIO18), pin#35(GPIO19), pin#38(GPIO20), pin#40(GPIO21), pin#15(GPIO22), pin#11(GPIO23), pin#18(GPIO24), pin#22(GPIO25), pin#37(GPIO26), pin#13(GPIO27)
   Minimun set:
   pin#29(GPIO5), pin#31(GPIO6), pin#33(GPIO13), pin#35(GPIO19), pin#37(GPIO26), pin#32(GPIO12), pin#36(GPIO16)

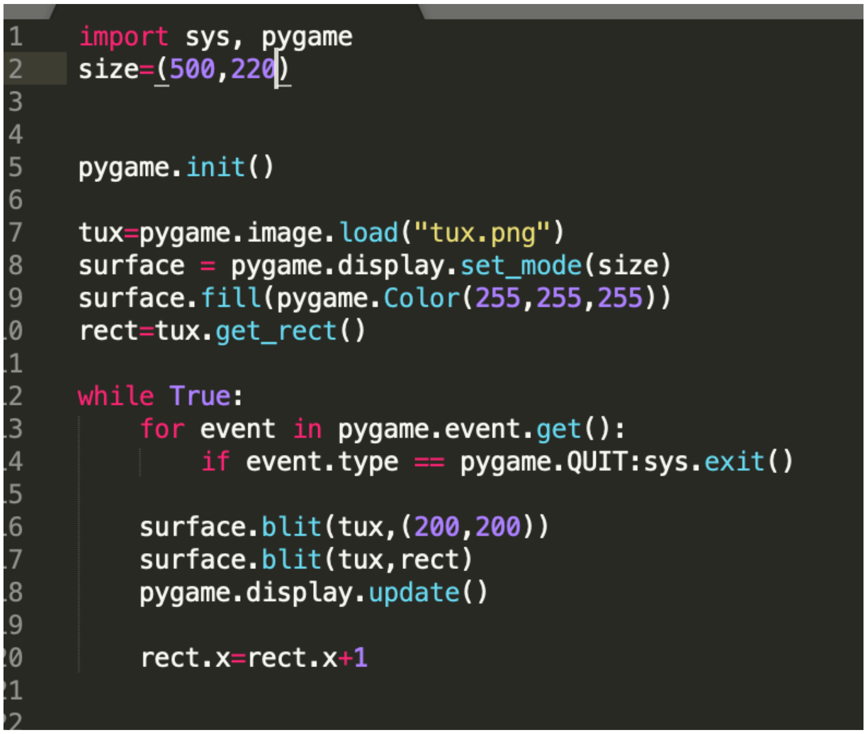
5. Here’s the correct order:
   The shell script should firstly call video_control.py and run it in background; then call mplayer to play video on foreground. Thus, mplayer can receive command

sent from video_control through video_fifo and return to command window when command “quit” is detected.
If we do it in reverse order, which means, call mplayer to play video on foreground and then call video_control.py to run it in background, we won’t be able to send command through FIFO because control process wil not start until Mplayer process ends.

6. When we connect Pi to the Internet via Ethernet or wifi, accurate time and date will update automatically from the global ntp(network time protocol) server according to our chosen timezone and country.
   If we want to keep accurate time and date when we cut off the internet or power off the Pi, we need extra Real Time Clock module.

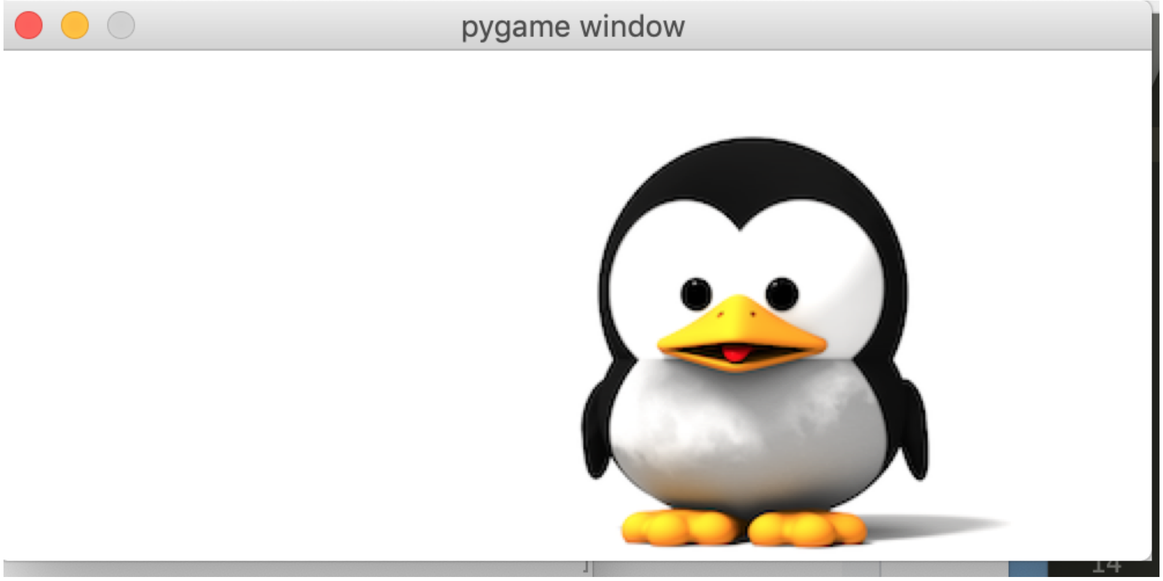
7. Linux is a multi-user, multi-process system, therefore, it allows two or more users modify the configuration at the same time. There could be situations when one user set the GPIO as input mode and another set it as output mode. In such case when button is pressed and there’s no resistor between ground and output GPIO, which means power and ground is shorted, GPIO will be damaged because of large current. Therefore, it’s necessary to put a current-limiting resistor, R2 in this case. Because GPIO output voltage is 3.3v, a 1K ohm resistor could limit current to 3.3mA, which is acceptable.

8. a. Pygame is a python wrapper and surface is like a piece of paper that could be created with this pygame wrapper by “pygame.display.set_mode()”. Whatever we do to the surface, results will display on the screen. Rect is an object in pygame used to store and manipulate rectangular areas, it can be created from objects that are already a rect or directly created from a combination of left, top, width and height values. Besides, we could use pygame.draw.rect() to draw a rectangular.
   b. Use surface and rect to animate image: firstly we set the size and background color of surface, load an image and use image.get_rect() to get the location information of image, change the display location and angle of image, such as changing rect.x or rect.y in the loop. By changing coordinate or other characteristic of the image, we could realize animating an image. If we want to animate by going through images, after erasing one image on the surface, surface display another image.
   Take the following program as an example:

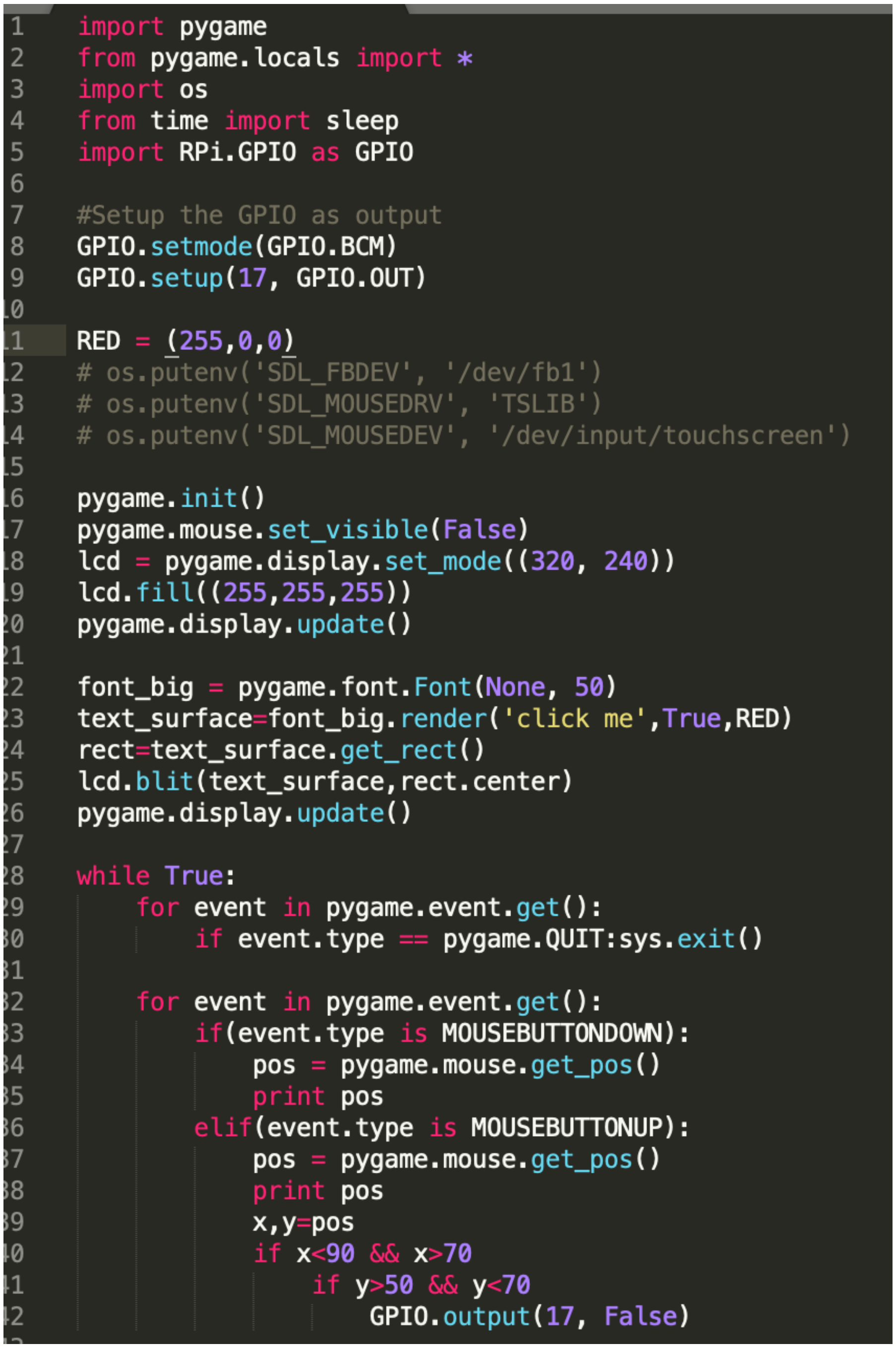


```
import sys, pygame
size=(500,220)


pygame.init()

tux=pygame.image.load("tux.png")
surface = pygame.display.set_mode(size)
surface.fill(pygame.Color(255,255,255))
rect=tux.get_rect()

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:sys.exit()

    surface.blit(tux,(200,200))
    surface.blit(tux,rect)
    pygame.display.update()

    rect.x=rect.x+1

```

Running this program, the tux could moving rightward on the surface.

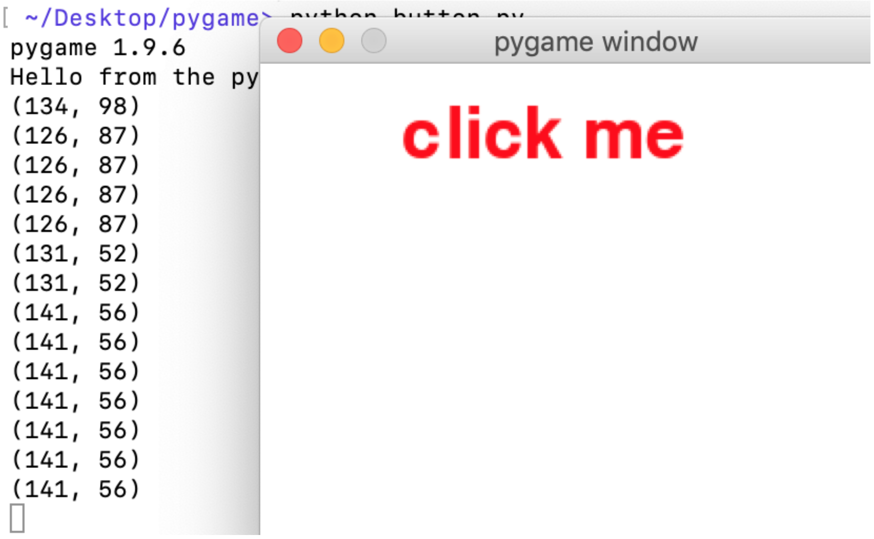


c. Establish a touchscreen button by detecting MOUSEBUTTONDOWN and MOUSEBUTTONUP events, program is as follows:



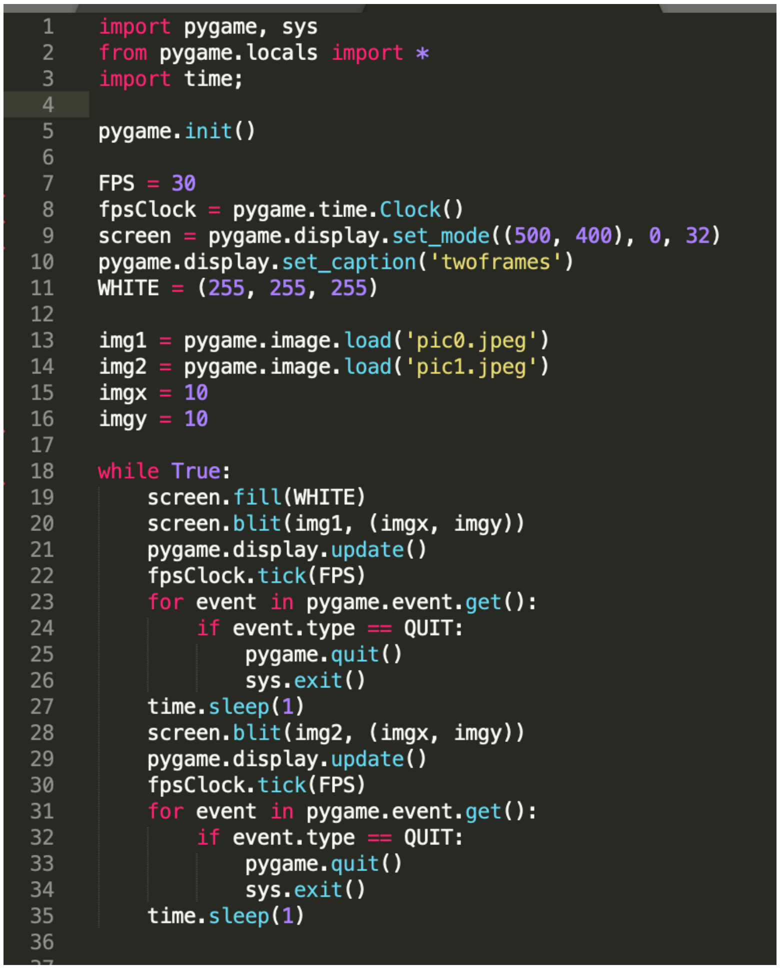
```
import pygame
from pygame.locals import *
import os
from time import sleep
import RPi.GPIO as GPIO

#Setup the GPIO as output
GPIO.setmode(GPIO.BCM)
GPIO.setup(17, GPIO.OUT)

RED = (255,0,0)
# os.putenv('SDL_FBDEV', '/dev/fb1')
# os.putenv('SDL_MOUSEDRV', 'TSLIB')
# os.putenv('SDL_MOUSEDEV', '/dev/input/touchscreen')

pygame.init()
pygame.mouse.set_visible(False)
lcd = pygame.display.set_mode((320, 240))
lcd.fill((255,255,255))
pygame.display.update()

font_big = pygame.font.Font(None, 50)
text_surface=font_big.render('click me',True,RED)
rect=text_surface.get_rect()
lcd.blit(text_surface,rect.center)
pygame.display.update()

while True:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:sys.exit()

    for event in pygame.event.get():
        if(event.type is MOUSEBUTTONDOWN):
            pos = pygame.mouse.get_pos()
            print pos
        elif(event.type is MOUSEBUTTONUP):
            pos = pygame.mouse.get_pos()
            print pos
            x,y=pos
            if x<90 && x>70
                if y>50 && y<70
                    GPIO.output(17, False)
```

I tried this result on laptop by clicking with mouse:



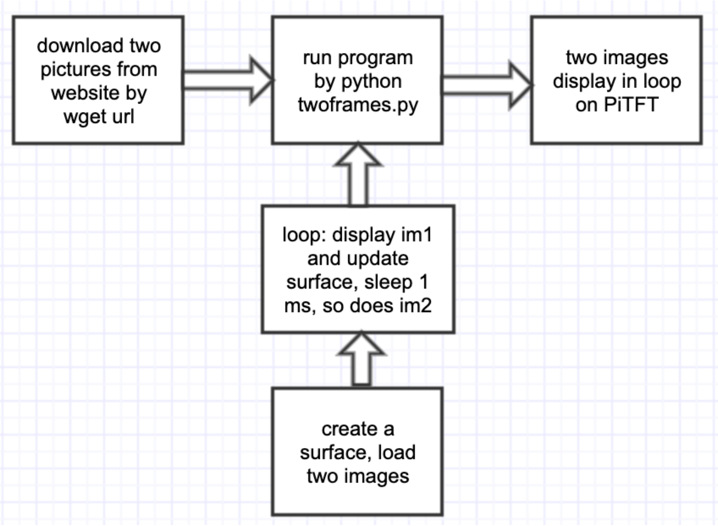
```
[ ~/Desktop/pygame> python button.py
pygame 1.9.6
Hello from the py
(134, 98)
(126, 87)
(126, 87)
(126, 87)
(126, 87)
(131, 52)
(131, 52)
(141, 56)
(141, 56)
(141, 56)
(141, 56)
(141, 56)
(141, 56)
(141, 56)
```

Display the process of animating two images:



```python
import pygame, sys
from pygame.locals import *
import time;

pygame.init()

FPS = 30
fpsClock = pygame.time.Clock()
screen = pygame.display.set_mode((500, 400), 0, 32)
pygame.display.set_caption('twoframes')
WHITE = (255, 255, 255)

img1 = pygame.image.load('pic0.jpeg')
img2 = pygame.image.load('pic1.jpeg')
imgx = 10
imgy = 10

while True:
    screen.fill(WHITE)
    screen.blit(img1, (imgx, imgy))
    pygame.display.update()
    fpsClock.tick(FPS)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
    time.sleep(1)
    screen.blit(img2, (imgx, imgy))
    pygame.display.update()
    fpsClock.tick(FPS)
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
    time.sleep(1)

```

Diagram:


download two pictures from website by wget url
run program by python twoframes.py
two images display in loop on PiTFT
loop: display im1 and update surface, sleep 1 ms, so does im2
create a surface, load two images

9. Polling is a CPU technique that keeps checking the state of external devices at regular intervals. The downside of polling is that it wastes CPU cycling and is very time-dependent, especially in case when detecting multiple buttons behaviour. Using callback, instead, is an interrput-handling mechanism. Peripheral devices send interrupt signal to get attention from CPU, which saves a lot of CPU cycling and time, and it is not time-dependent. Besides, using callback can receive interrupts from different In button case, GPIO edge detection will send interrupt signal to CPU to trigger python on an event.
Interrupt controller is used in processor when executing a callback routine.

10. If we don't first create video_fifo, when we press a button, video control.py cannot find the fifo file, so it will create a file named video_fifo(but not actual fifo!). The mplayer will not respond to any button and continuously play the video. If video_fifo is created correctly at first, mplayer will receive commands from fifo file and respond correctly.