```python
# import packages
import pygame
from pygame.locals import*
import os
import RPi.GPIO as GPIO
import subprocess
import random
import time
import cv2
import numpy as np
import cv2 as cv

# set up piTFT touchscreen
#os.putenv('SDL_VIDEODRIVER','fbcon')
#os.putenv('SDL_FBDEV','/dev/fb0')
#os.putenv('SDL_MOUSEDRV','TSLIB')
#os.putenv('SDL_MOUSEDEV','/dev/input/touchscreen')

pygame.init()
#pygame.mouse.set_visible(False)

# set up quit button
GPIO.setmode(GPIO.BCM)
GPIO.setup(17, GPIO.IN, pull_up_down=GPIO.PUD_UP)

def GPIO17_call_back(channel):
    global code_running
    global color_selection
    global select
    code_running=False
    color_selection=False
    select=False
    start=False
    sort=False

GPIO.add_event_detect(17, GPIO.FALLING, callback=GPIO17_call_back,bouncetime=300)

# motor rotation set up (first motor controller)
GPIO.setup(5,GPIO.OUT)
GPIO.setup(6,GPIO.OUT)
GPIO.setup(13,GPIO.OUT) # PWM A

# track motor set up (first motor controller)
GPIO.setup(20,GPIO.OUT)
GPIO.setup(21,GPIO.OUT)
GPIO.setup(16,GPIO.OUT) # PWM B

# led set up (second motor controller)
GPIO.setup(25,GPIO.OUT)
GPIO.setup(23,GPIO.OUT)
GPIO.setup(24,GPIO.OUT) # PWM A

# track motor set up (second motor controller)
GPIO.setup(22,GPIO.OUT)
GPIO.setup(4,GPIO.OUT)
GPIO.setup(27,GPIO.OUT) # PWM B

# color set up
WHITE=255,255,255
BLACK=0,0,0
RED = 230,0,0
GREEN = 0, 200, 0
GREEN_START = 35,150,10

# screen setup
screen = pygame.display.set_mode((320,240))
screen.fill(BLACK)
my_font=pygame.font.Font(None,25)
my_font2=pygame.font.Font(None,20)
screen.fill(BLACK)
my_text=""
text_pos= [0,0]

# import basket image
basket_image=pygame.image.load("laundry_basket.png")
basket_image=pygame.transform.scale(basket_image, (110,110))

def basket(x,y):
    screen.blit(basket_image,(x,y))

# initalize variables
code_running=True
color_selection=True
select=True
sort=False
start=True
starttime=time.time()

# basket 1 initialization
choose_white1=False
```

```
choose_color1=False
choose_red1=False
choose_light1=False
choose_dark1=False

#basket 2 initialization
choose_white2=False
choose_color2=False
choose_red2=False
choose_light2=False
choose_dark2=False

#basket 3 initialization
choose_white3=False
choose_color3=False
choose_red3=False
choose_light3=False
choose_dark3=False

# clothing color initialization
CLOTHING_RED=False
CLOTHING_COLOR=False
CLOTHING_WHITE=False
CLOTHING_DARK=False
CLOTHING_LIGHT=False

# red:
light_red1=np.array([0,51,20])
dark_red1=np.array([15,255,255])
light_red2=np.array([160,51,20])
dark_red2=np.array([180,255,255])

# white:
#light_white=np.array([0,0,225])
#dark_white=np.array([180,30,255])
light_white=np.array([0,0,200])
dark_white=np.array([180,80,255])

# darks:
light_dark1=np.array([0,100,0])
dark_dark1=np.array([180,255,100])
light_dark2=np.array([110,150,0])
dark_dark2=np.array([160,255,100])

# lights:
```

```python
light_light=np.array([0,30,200])
dark_light=np.array([180,100,225])

# colors:
#light_color=np.array([0,100,100])
#dark_color=np.array([180,225,225])
light_color=np.array([0,75,75])
dark_color=np.array([180,175,175])

# Initialize motors
p1=GPIO.PWM(13,50)
p2=GPIO.PWM(16,50)
p_led=GPIO.PWM(24,50)
p_rot=GPIO.PWM(27,50)
p1.start(0)
p2.start(0)
p_rot.start(0)
p_led.start(0)

# video capture source camera setup
#cap = cv2.VideoCapture(0)

# led direction
GPIO.output(23,GPIO.LOW)
GPIO.output(25,GPIO.HIGH)

# motor direction

def basket1():
    # motor rotate
    print("rotate")
    GPIO.output(4,GPIO.LOW)
    GPIO.output(22,GPIO.HIGH)
    p_rot.ChangeDutyCycle(80)
    time.sleep(0.25)
    p_rot.ChangeDutyCycle(0)

    # motor forward
    print("forward")
    GPIO.output(6,GPIO.HIGH)
    GPIO.output(5,GPIO.LOW)
    GPIO.output(20,GPIO.HIGH)
    GPIO.output(21,GPIO.LOW)
    p1.ChangeDutyCycle(100)
    p2.ChangeDutyCycle(100)
    time.sleep(0.7)

    # motor pause
    p1.ChangeDutyCycle(0)
    p2.ChangeDutyCycle(0)
    time.sleep(0.2)

    # motor back
    print("back")
    GPIO.output(5,GPIO.HIGH)
    GPIO.output(6,GPIO.LOW)
    GPIO.output(21,GPIO.HIGH)
    GPIO.output(20,GPIO.LOW)
    p1.ChangeDutyCycle(85)
    p2.ChangeDutyCycle(85)
    time.sleep(1.1)
    p1.ChangeDutyCycle(0)
    p2.ChangeDutyCycle(0)
    time.sleep(0.2)

    # motor rotate
    print("rotate back")
    GPIO.output(22,GPIO.LOW)
    GPIO.output(4,GPIO.HIGH)
    p_rot.ChangeDutyCycle(80)
    time.sleep(0.25)
    p_rot.ChangeDutyCycle(0)

def basket2():
    # motor forward
    print("forward")
    GPIO.output(6,GPIO.HIGH)
    GPIO.output(5,GPIO.LOW)
    GPIO.output(20,GPIO.HIGH)
    GPIO.output(21,GPIO.LOW)
    p1.ChangeDutyCycle(100)
    p2.ChangeDutyCycle(100)
    time.sleep(0.75)

    # motor pause
    p1.ChangeDutyCycle(0)
    p2.ChangeDutyCycle(0)
    time.sleep(0.2)

    # motor back
```

```
        print("back")
        GPIO.output(5,GPIO.HIGH)
        GPIO.output(6,GPIO.LOW)
        GPIO.output(21,GPIO.HIGH)
        GPIO.output(20,GPIO.LOW)
        p1.ChangeDutyCycle(85)
        p2.ChangeDutyCycle(85)
        time.sleep(1.1)
        p1.ChangeDutyCycle(0)
        p2.ChangeDutyCycle(0)
        time.sleep(0.2)

    def basket3():

        # motor rotate
        print("rotate")
        GPIO.output(22,GPIO.LOW)
        GPIO.output(4,GPIO.HIGH)
        p_rot.ChangeDutyCycle(80)
        time.sleep(0.25)
        p_rot.ChangeDutyCycle(0)

        # motor forward
        print("forward")
        GPIO.output(6,GPIO.HIGH)
        GPIO.output(5,GPIO.LOW)
        GPIO.output(20,GPIO.HIGH)
        GPIO.output(21,GPIO.LOW)
        p1.ChangeDutyCycle(100)
        p2.ChangeDutyCycle(100)
        time.sleep(0.75)

        # motor pause
        p1.ChangeDutyCycle(0)
        p2.ChangeDutyCycle(0)
        time.sleep(0.2)

        # motor back
        print("back")
        GPIO.output(5,GPIO.HIGH)
        GPIO.output(6,GPIO.LOW)
        GPIO.output(21,GPIO.HIGH)
        GPIO.output(20,GPIO.LOW)
        p1.ChangeDutyCycle(85)
        p2.ChangeDutyCycle(85)
        time.sleep(1.1)
        p1.ChangeDutyCycle(0)
        p2.ChangeDutyCycle(0)
        time.sleep(0.2)

        # motor rotate
        print("rotate back")
        GPIO.output(4,GPIO.LOW)
        GPIO.output(22,GPIO.HIGH)
        p_rot.ChangeDutyCycle(80)
        time.sleep(0.25)
        p_rot.ChangeDutyCycle(0)

    def okbutton():
        pygame.draw.circle(screen,GREEN_START,(260,200),25)
        my_buttons={'OK':(260,200)}
        for my_text,text_pos, in my_buttons.items():
            text_surface=my_font.render(my_text,True,WHITE)
            rect=text_surface.get_rect(center=text_pos)
            screen.blit(text_surface,rect)

    def whitebutton(white_input):
        if white_input==True:
            pygame.draw.rect(screen,GREEN,pygame.Rect(20,80,80,30))
        else:
            pygame.draw.rect(screen,RED,pygame.Rect(20,80,80,30))

        for my_text,text_pos, in my_buttons.items():
            text_surface=my_font.render("Whites",True,WHITE)
            rect=text_surface.get_rect(center=(60,95))
            screen.blit(text_surface,rect)
        pygame.display.flip()

    def colorbutton(color_input):
        if color_input==True:
            pygame.draw.rect(screen,GREEN,pygame.Rect(120,80,80,30))
        else:
            pygame.draw.rect(screen,RED,pygame.Rect(120,80,80,30))

        for my_text,text_pos, in my_buttons.items():
            text_surface=my_font.render("Colors",True,WHITE)
            rect=text_surface.get_rect(center=(160,95))
            screen.blit(text_surface,rect)
        pygame.display.flip()

```

```
def redbutton(red_input):
    if red_input==True:
        pygame.draw.rect(screen,GREEN,pygame.Rect(220,80,80,30))
    else:
        pygame.draw.rect(screen,RED,pygame.Rect(220,80,80,30))

    for my_text,text_pos, in my_buttons.items():
        text_surface=my_font.render("Reds",True,WHITE)
        rect=text_surface.get_rect(center=(260,95))
        screen.blit(text_surface,rect)
    pygame.display.flip()

def lightbutton(light_input):
    if light_input==True:
        pygame.draw.rect(screen,GREEN,pygame.Rect(70,130,80,30))
    else:
        pygame.draw.rect(screen,RED,pygame.Rect(70,130,80,30))

    for my_text,text_pos, in my_buttons.items():
        text_surface=my_font.render("Lights",True,WHITE)
        rect=text_surface.get_rect(center=(110,145))
        screen.blit(text_surface,rect)
    pygame.display.flip()

def darkbutton(dark_input):
    if dark_input==True:
        pygame.draw.rect(screen,GREEN,pygame.Rect(170,130,80,30))
    else:
        pygame.draw.rect(screen,RED,pygame.Rect(170,130,80,30))

    for my_text,text_pos, in my_buttons.items():
        text_surface=my_font.render("Darks",True,WHITE)
        rect=text_surface.get_rect(center=(210,145))
        screen.blit(text_surface,rect)
    pygame.display.flip()

# main code
while code_running:

    time.sleep(0.1)
    color_selection = True
    start=True

    screen.fill(BLACK)

    # initalize screen
    text_surface=my_font.render("Select basket below:",True,WHITE)
    rect=text_surface.get_rect(center=(160,30))
    screen.blit(text_surface,rect)

    basket(13,47)
    basket(105,47)
    basket(195,47)

    pygame.draw.circle(screen,GREEN_START,(260,200),25)
    my_buttons={'start':(260,200)}
    for my_text,text_pos, in my_buttons.items():
        text_surface=my_font.render(my_text,True,WHITE)
        rect=text_surface.get_rect(center=text_pos)
        screen.blit(text_surface,rect)


    my_buttons={'Basket 1':(70,155), 'Basket 2':(160,155), 'Basket 3': (250,155)}
    for my_text,text_pos, in my_buttons.items():
        text_surface=my_font2.render(my_text,True,WHITE)
        rect=text_surface.get_rect(center=text_pos)
        screen.blit(text_surface,rect)

    pygame.display.flip()

#################### select basket ############################

    for event in pygame.event.get():
        if(event.type is MOUSEBUTTONDOWN):
            pos=pygame.mouse.get_pos()
            x,y=pos
        elif(event.type is MOUSEBUTTONUP):

            time.sleep(0.1)
            while color_selection and start:

                time.sleep(0.1)

                ################# basket 1 ################
                if (y> 60 and y<180 and x<105):

                    time.sleep(0.1)
                    select=True

                    screen.fill(BLACK)
                    okbutton()

                    text_surface=my_font.render("Basket 1:",True,WHITE)
                    rect=text_surface.get_rect(center=(160,40))
                    screen.blit(text_surface,rect)

                    pygame.display.flip()

                    # color selection
                    while select:

                        # white button
                        whitebutton(choose_white1)

                        # color buttton
                        colorbutton(choose_color1)

                        # red buttton
                        redbutton(choose_red1)

                        # light button
                        lightbutton(choose_light1)

                        # dark button
                        darkbutton(choose_dark1)

                        for event in pygame.event.get():
                            if(event.type is MOUSEBUTTONDOWN):
                                pos=pygame.mouse.get_pos()
                                m,n=pos
                            elif(event.type is MOUSEBUTTONUP):

                                if (m>20 and m<=100 and n>80 and n <=110):
                                    choose_white1 = not choose_white1

                                elif (m>120 and m<=200 and n>80 and n<=110):
                                    choose_color1 = not choose_color1

                                elif (m>220 and m<=300 and n>80 and n<=110):
                                    choose_red1 = not choose_red1

                                elif (m>70 and m<=150 and n>130 and n<=160):
                                    choose_light1 = not
```

```
                                        choose_light1

                                    elif (m>170 and m<=250 and n>130 and n<=160):
                                        choose_dark1 = not choose_dark1

                                    # okay
                                    elif (m>200 and n>17):
                                        select = False
                                        color_selection=False


                    ################# basket 2 ################
                    elif (y>60 and y<180 and x>=105 and x<215):

                        time.sleep(0.1)
                        select=True

                        screen.fill(BLACK)
                        okbutton()

                        text_surface=my_font.render("Basket 2:",True,WHITE)
                        rect=text_surface.get_rect(center=(160,40))
                        screen.blit(text_surface,rect)

                        pygame.display.flip()

                        # color selection
                        while select:

                            # white button
                            whitebutton(choose_white2)

                            # color buttton
                            colorbutton(choose_color2)

                            # red buttton
                            redbutton(choose_red2)

                            # light button
                            lightbutton(choose_light2)

                            # dark button
                            darkbutton(choose_dark2)

                            for event in pygame.event.get():
```

```python
                            if(event.type is MOUSEBUTTONDOWN):
                                pos=pygame.mouse.get_pos()
                                m,n=pos
                            elif(event.type is MOUSEBUTTONUP):

                                if (m>20 and m<=100 and n>80 and n <=110):
                                    choose_white2 = not choose_white2

                                elif (m>120 and m<=200 and n>80 and n<=110):
                                    choose_color2 = not choose_color2

                                elif (m>220 and m<=300 and n>80 and n<=110):
                                    choose_red2 = not choose_red2

                                elif (m>70 and m<=150 and n>130 and n<=160):
                                    choose_light2 = not choose_light2

                                elif (m>170 and m<=250 and n>130 and n<=160):
                                    choose_dark2 = not choose_dark2

                                # okay
                                elif (m>200 and n>170):
                                    select = False
                                    color_selection=False


                ################# basket 3 ################
                elif (y>60 and y<=180 and x>=215):

                    time.sleep(0.1)
                    select=True

                    screen.fill(BLACK)
                    okbutton()

                    text_surface=my_font.render("Basket 3:",True,WHITE)
                    rect=text_surface.get_rect(center=(160,40))
                    screen.blit(text_surface,rect)

                    pygame.display.flip()

                    # color selection
                    while select:

                        # white button
                        whitebutton(choose_white3)

                        # color buttton
                        colorbutton(choose_color3)

                        # red buttton
                        redbutton(choose_red3)

                        # light button
                        lightbutton(choose_light3)

                        # dark button
                        darkbutton(choose_dark3)

                        for event in pygame.event.get():
                            if(event.type is MOUSEBUTTONDOWN):
                                pos=pygame.mouse.get_pos()
                                m,n=pos
                            elif(event.type is MOUSEBUTTONUP):

                                if (m>20 and m<=100 and n>80 and n <=110):
                                    choose_white3 = not choose_white3

                                elif (m>120 and m<=200 and n>80 and n<=110):
                                    choose_color3 = not choose_color3

                                elif (m>220 and m<=300 and n>80 and n<=110):
                                    choose_red3 = not choose_red3

                                elif (m>70 and m<=150 and n>130 and n<=160):
                                    choose_light3 = not choose_light3

                                elif (m>170 and m<=250 and n>130 and n<=160):
                                    choose_dark3 = not choose_dark3

                                # okay
                                elif (m>200 and n>170):
                                    select = False
                                    color_selection=False
```

```

############################## start ###########################################
                elif (y>180 and x>215):

                    time.sleep(0.1)

                    cap = cv2.VideoCapture(0) #video capture source camera

                    sort = True

                    CLOTHING_RED=False
                    CLOTHING_COLOR=False
                    CLOTHING_WHITE=False
                    CLOTHING_DARK=False
                    CLOTHING_LIGHT=False

                    # turn light on
                    p_led.start(100)
                    time.sleep(0.5)
                    print("light on")

                    ret,frame = cap.read()
                    print("picture taken")

                    # turn off light
                    time.sleep(0.5)
                    p_led.ChangeDutyCycle(0)

                    # convert RBG to HSV
                    hsv=cv2.cvtColor(frame,cv2.COLOR_BGR2HSV)

                    # find colors
                    mask_red1=cv2.inRange(hsv,light_red1,dark_red1)
                    mask_red2=cv2.inRange(hsv,light_red2,dark_red2)
                    mask_white=cv2.inRange(hsv,light_white,dark_white)
                    mask_dark1=cv2.inRange(hsv,light_dark1,dark_dark1)
                    mask_dark2=cv2.inRange(hsv,light_dark2,dark_dark2)
                    mask_light=cv2.inRange(hsv,light_light,dark_light)
                    mask_color=cv2.inRange(hsv,light_color,dark_color)

                    # output
                    red1=cv2.bitwise_and(frame,frame,mask=mask_red1)
                    red2=cv2.bitwise_and(frame,frame,mask=mask_red2)
                    white=cv2.bitwise_and(frame,frame,mask=mask_white)
                    dark1=cv2.bitwise_and(frame,frame,mask=mask_dark1)
```

```
                        dark2=cv2.bitwise_and(frame,frame,mask=mask_dark2)
                        light=cv2.bitwise_and(frame,frame,mask=mask_light)
                        color=cv2.bitwise_and(frame,frame,mask=mask_color)

                        # Find area
                        area_red1 = 0
                        area_red2 = 0
                        area_white = 0
                        area_dark1 = 0
                        area_dark2 = 0
                        area_light = 0
                        area_color = 0
                        area_red = 0
                        area_dark = 0

                        # Remove noise
                        kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3,3))
                        opening_red1 = cv2.morphologyEx(mask_red1, cv2.MORPH_OPEN, kernel, iterations=1)
                        opening_red2 = cv2.morphologyEx(mask_red2, cv2.MORPH_OPEN, kernel, iterations=1)
                        opening_white = cv2.morphologyEx(mask_white, cv2.MORPH_OPEN, kernel, iterations=1)
                        opening_dark1 = cv2.morphologyEx(mask_dark1, cv2.MORPH_OPEN, kernel, iterations=1)
                        opening_dark2 = cv2.morphologyEx(mask_dark2, cv2.MORPH_OPEN, kernel, iterations=1)
                        opening_light = cv2.morphologyEx(mask_light, cv2.MORPH_OPEN, kernel, iterations=1)
                        opening_color = cv2.morphologyEx(mask_color, cv2.MORPH_OPEN, kernel, iterations=1)

                        # Find contours
                        original = frame.copy()
                        cnts_red1 = cv2.findContours(opening_red1, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
                        cnts_red1 = cnts_red1[0] if len(cnts_red1) == 2 else cnts_red1[1]
                        cnts_red2 = cv2.findContours(opening_red2, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
                        cnts_red2 = cnts_red2[0] if len(cnts_red2) == 2 else cnts_red2[1]
                        cnts_white = cv2.findContours(opening_white, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
                        cnts_white = cnts_white[0] if len(cnts_white) == 2 else cnts_white[1]
                        cnts_dark1 = cv2.findContours(opening_dark1, cv2.RETR_EXTERNAL,
                        cv2.CHAIN_APPROX_SIMPLE)
                        cnts_dark1 = cnts_dark1[0] if len(cnts_dark1) == 2 else cnts_dark1[1]
                        cnts_dark2 = cv2.findContours(opening_dark2, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
                        cnts_dark2= cnts_dark2[0] if len(cnts_dark2) == 2 else cnts_dark2[1]
                        cnts_light = cv2.findContours(opening_light, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
                        cnts_light = cnts_light[0] if len(cnts_light) == 2 else cnts_light[1]
                        cnts_color = cv2.findContours(opening_color, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
                        cnts_color = cnts_color[0] if len(cnts_color) == 2 else cnts_color[1]

                        for c in cnts_red1:
                            area_red1 += cv2.contourArea(c)
                            cv2.drawContours(original,[c], 0, (0,0,0), 2)
                        for c in cnts_red2:
                            area_red2 += cv2.contourArea(c)
                            cv2.drawContours(original,[c], 0, (0,0,0), 2)
                        for c in cnts_white:
                            area_white += cv2.contourArea(c)
                            cv2.drawContours(original,[c], 0, (0,0,0), 2)
                        for c in cnts_dark1:
                            area_dark1 += cv2.contourArea(c)
                            cv2.drawContours(original,[c], 0, (0,0,0), 2)
                        for c in cnts_dark2:
                            area_dark2 += cv2.contourArea(c)
                            cv2.drawContours(original,[c], 0, (0,0,0), 2)
                        for c in cnts_light:
                            area_light += cv2.contourArea(c)
                            cv2.drawContours(original,[c], 0, (0,0,0), 2)
                        for c in cnts_color:
                            area_color += cv2.contourArea(c)
                            cv2.drawContours(original,[c], 0, (0,0,0), 2)

                        # combine red and darks
                        area_red=area_red1+area_red2
                        area_dark=area_dark1+area_dark2

                        # Determine dominant color
                        if area_red>area_white and area_red>area_dark and
```

```python
                    area_red>area_light and area_red>area_color:
                        print('ning smells like cherries')
                        CLOTHING_RED = True
                    elif area_white>area_red and area_white>area_dark and area_white>area_light and area_white>area_color:
                        print('ning smells like a cloud')
                        CLOTHING_WHITE = True
                    elif area_dark>area_red and area_dark>area_white and area_dark>area_light and area_dark>area_color:
                        print('ning smells like dark chocolate')
                        CLOTHING_DARK = True
                    elif area_light>area_dark and area_light>area_red and area_light>area_white and area_light>area_color:
                        print('ning smells like an LED')
                        CLOTHING_LIGHT = True
                    else:
                        print('ning smells like a lucky charms')
                        CLOTHING_COLOR = True


                    cv2.imshow('img1',frame)

                    #cv2.waitKey(0)
                    time.sleep(1)
                    cap.release()
                    cv2.destroyAllWindows()

                    ########### analyze color detection / code motors ##############
                    while sort:

                        ############ red ############
                        if CLOTHING_RED==True and choose_red1==True:
                            print("basket 1: red")

                            basket1()
                            sort=False
                            start=False

                        elif CLOTHING_RED==True and choose_red2==True:

                            basket2()
                            sort=False
                            start=False

                        elif CLOTHING_RED==True and choose_red3==True:

                            basket3()
                            sort=False
                            start=False

                        ############ color ############
                        if CLOTHING_COLOR==True and choose_color1==True:
                            print("basket 1: color")

                            basket1()
                            sort=False
                            start=False

                        elif CLOTHING_COLOR==True and choose_color2==True:

                            basket2()
                            sort=False
                            start=False

                        elif CLOTHING_COLOR==True and choose_color3==True:

                            basket3()
                            sort=False
                            start=False

                        ############ WHITE ############
                        if CLOTHING_WHITE==True and choose_white1==True:
                            print("basket 1: white")

                            basket1()
                            sort=False
                            start=False

                        elif CLOTHING_WHITE==True and choose_white2==True:

                            basket2()
                            sort=False
                            start=False

                        elif CLOTHING_WHITE==True and choose_white3==True:

                            basket3()
                            sort=False
                            start=False

                        ############ dark ############
                        if CLOTHING_DARK==True and choose_dark1==True:
                            print("basket 1: dark")

                            basket1()
                            sort=False
                            start=False

                        elif CLOTHING_DARK==True and choose_dark2==True:

                            basket2()
                            sort=False
                            start=False

                        elif CLOTHING_DARK==True and choose_dark3==True:

                            basket3()
                            sort=False
                            start=False

                        ############ lights ############
                        if CLOTHING_LIGHT==True and choose_light1==True:

                            print("basket 1: light")
                            basket1()
                            sort=False
                            start=False

                        elif CLOTHING_LIGHT==True and choose_light2==True:

                            basket2()
                            sort=False
                            start=False

                        elif CLOTHING_LIGHT==True and choose_light3==True:

                            basket3()
                            sort=False
                            start=False


GPIO.cleanup()
```