# **Program 1: Food Delivery Program  Report**

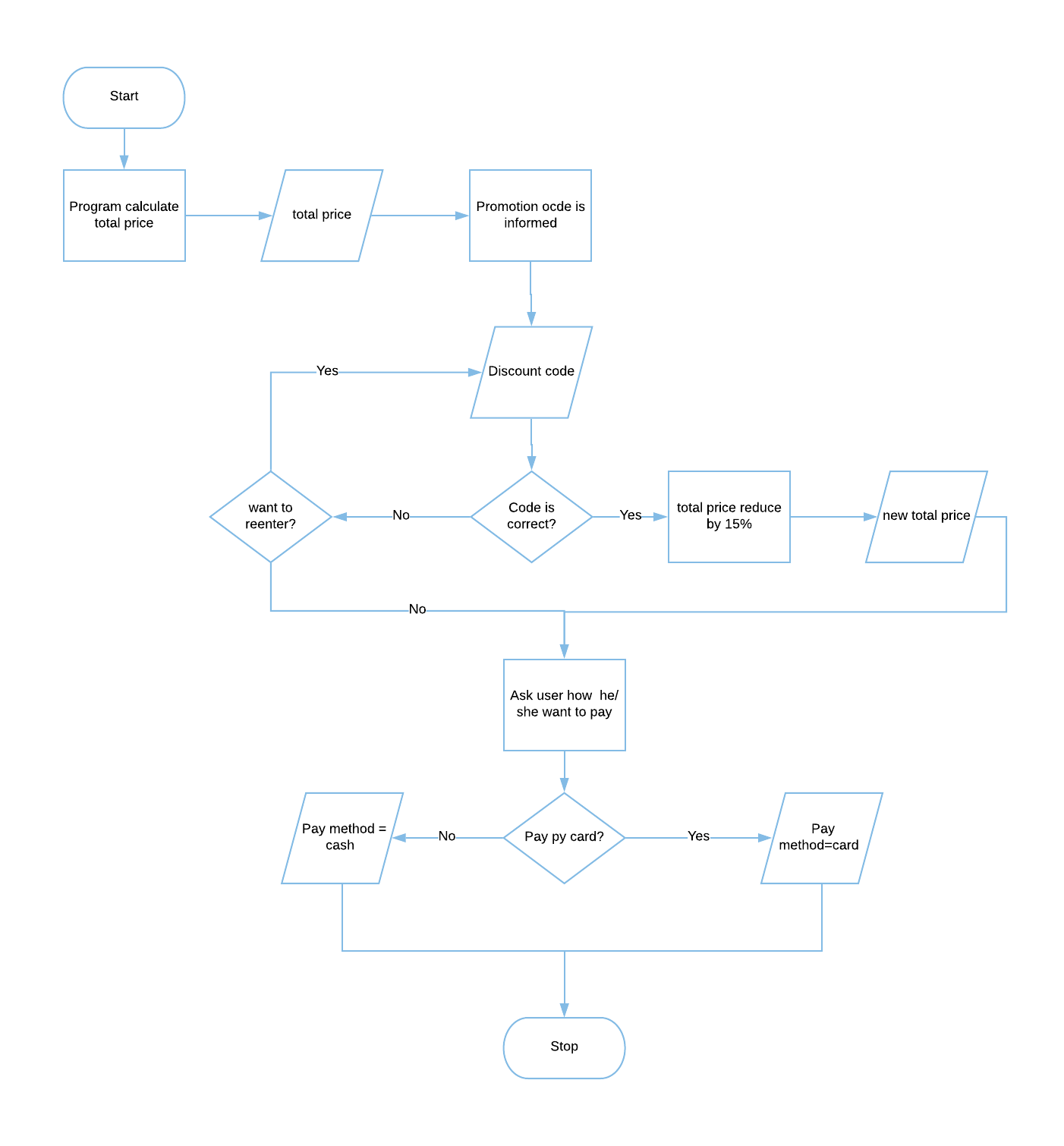
# Food delivery program flow chart







# Subprocess: Payment



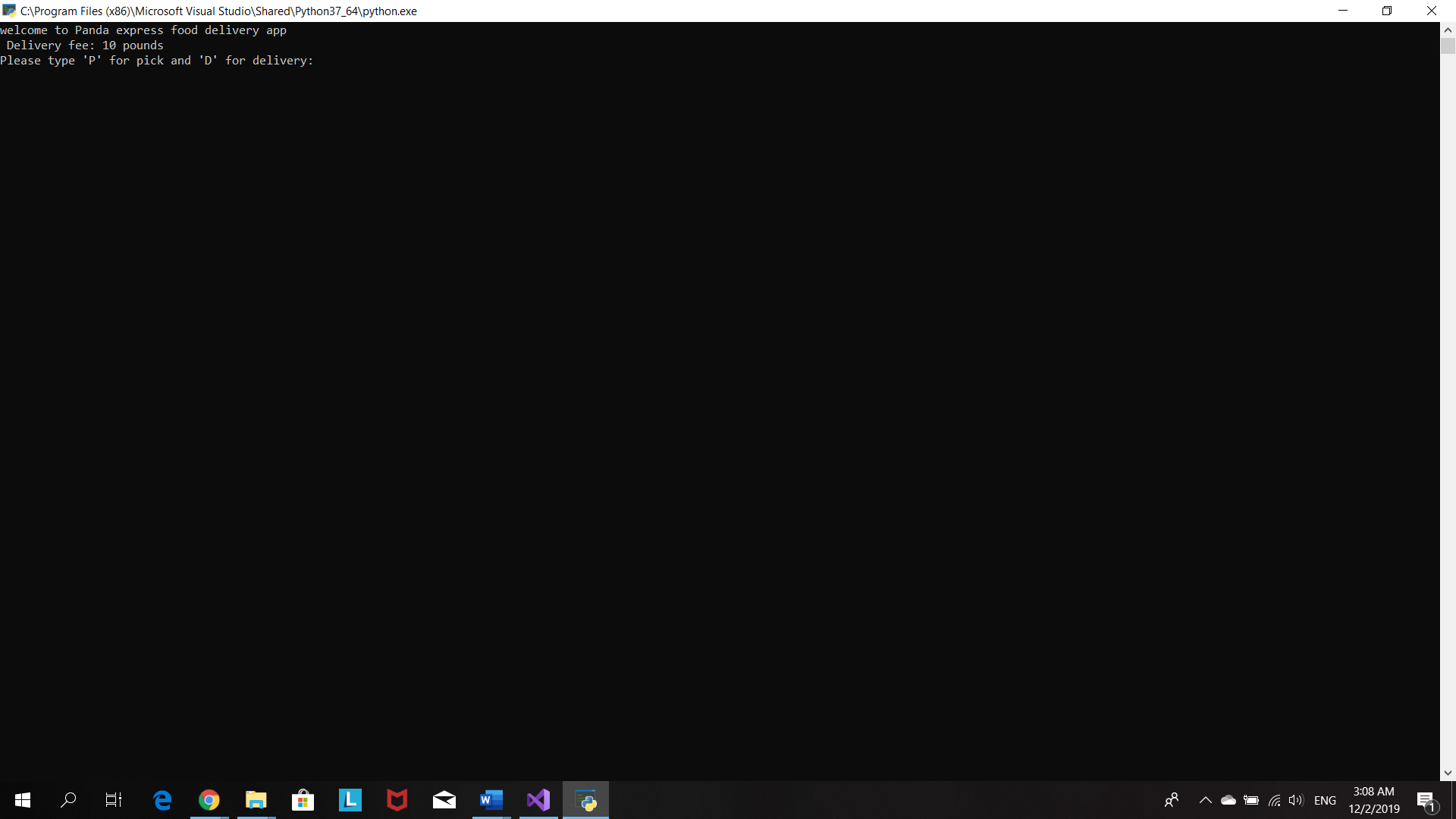
# Testing Table- Program 1: Food Delivery Program

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. | Item to test | Test description | Test input | Expected result | Actual result | Comment |
| 1 | input("Please type 'P' for  pick and 'D' for delivery: ") | Typical valid data | P | Value accepted | Value accepted-program continue as normal | - |
| 2 | input("Please type 'P' for  pick and 'D' for delivery: ") | Extreme valid data | d | Value accepted | Value accepted-program continue as normal | - |
| 3 | input("Please type 'P' for  pick and 'D' for delivery: ") | Invalid data | T | Error massage: re-enter value | Value accepted -program continue but skip the rest of function pickupDelivery | Should let user input a correct data again |
| 4 | input("Please type 'P' for  pick and 'D' for delivery: ") | Invalid extreme data | i | Error massage: re-enter value | Value accepted -program continue but skip the rest of function pickupDelivery | Should let user input a correct value again |
| 5 | input("Please type 'P' for  pick and 'D' for delivery: ") | Erroneous data | 100 | Error massage: re-enter value | Value accepted -program continue but skip the rest of function pickupDelivery | Should let user input a correct value again |
| 6 | input("Please type 'P' for  pick and 'D' for delivery: ") | Erroneous data | $%^&(() | Error massage: re-enter value | Value accepted -program continue but skip the rest of function pickupDelivery | Should let user input a correct value again |
| 7 | input("Costumer Name: ") | Typical valid data | Jay | Value accepted | Value accepted-program continue as normal | - |
| 8 | input("Costumer Name: ") | Extreme valid data | ppppppppppppppppppp | Value accepted | Value accepted-program continue as normal |  |
| 9 | input("Costumer Name: ") | Invalid data | 10 | Error massage: re-enter value | Value accepted-program continue as normal | Should let user input a correct value again |
| 10 | input("Costumer Name: ") | Invalid extreme data | -810 | Error massage: re-enter value | Value accepted-program continue as normal | Should let user input a correct value again |
| 11 | input("Costumer Name: ") | Erroneous data | $%^&(() | Error massage: re-enter value | Value accepted-program continue as normal | Should let user input a correct value again |
| 12 | input("Costumer Name: ") | Erroneous data | “HELP” | Error massage:  Re-enter value | Value accepted-program continue as normal | Should let user input a correct value again |
| 13 | input("Please enter you house number: ") | Typical valid data | 211 | Value accepted | Value accepted-program continue as normal | - |
| 14 | input("Please enter you house number: ") | Extreme valid data | 2000000 | Error massage: re-enter value | Value accepted-program continue as normal | Limit number of house number |
| 15 | input("Please enter you house number: ") | Invalid data | T | Error massage: re-enter value | Value accepted-program continue as normal | Allow user to input only number |
| 16 | input("Please enter you house number: ") | Extreme valid data | pppppppppppppppppppppppppppppppp | Error massage: re-enter value | Value accepted-program continue as normal | Allow user to input only number |
| 17 | input("Please enter you house number: ") | Erroneous data | $%^& | Error massage: re-enter value | Value accepted-program continue as normal | Allow user to input only number |
| 18 | input("Please enter you house number: ") | Erroneous data | “HELP!!!” | Error massage: re-enter value | Value accepted-program continue as normal | Allow user to input only number |
| 19 | street=input("Please enter street: ") | Typical valid data | Baker St | Value accepted | Value accepted-program continue as normal | - |
| 20 | street=input("Please enter street: ") | Extreme valid data | qwertyuiopdsfvhvjsdjsd St | Error massage: re-enter value | Value accepted-program continue as normal | Limit number or input character |
| 21 | street=input("Please enter street: ") | Invalid data | 10 | Error massage: re-enter value | Value accepted-program continue as normal | Allow user to input only letter and dot |
| 22 | street=input("Please enter street: ") | Invalid extreme data | -785 | Error massage: re-enter value | Value accepted-program continue as normal | Allow user to input only letter and dot |
| 23 | street=input("Please enter street: ") | Erroneous data | $£%^&\* | Error massage: re-enter value | Value accepted-program continue as normal | Allow user to input only letter and dot |
| 24 | street=input("Please enter street: ") | Erroneous data | “Sherlock Holmes wow!!!!” | Error massage: re-enter value | Value accepted-program continue as normal | Allow user to input only letter and dot |
| 25 | postcode=input("Please  enter postcode: ") | Typical valid data | NW1 6XE | Value accepted | Value accepted-program continue as normal | - |
| 26 | postcode=input("Please  enter postcode: ") | Extreme valid data | NW120 6XE | Error massage: re-enter value | Value accepted-program continue as normal | Limit amount of character can be inputted |
| 27 | postcode=input("Please  enter postcode: ") | Invalid data | 64545 | Error massage: re-enter value | Value accepted-program continue as normal | Should let user input a correct value again |
| 28 | postcode=input("Please  enter postcode: ") | Invalid extreme data | --888888 | Error massage: re-enter value | Value accepted-program continue as normal | Limit amount of character can be inputted |
| 29 | postcode=input("Please  enter postcode: ") | Erroneous data | $%^% | Error massage: re-enter value | Value accepted-program continue as normal | Should let user input a correct value again |
| 30 | postcode=input("Please  enter postcode: ") | Erroneous data | £$%^&\*(093-406834-0683 | Error massage: re-enter value | Value accepted-program continue as normal | Limit amount of character can be inputted |
| 31 | input("Do you want your order to deliver in specific time? (Y/N): ") | Typical valid data | Y | Value accepted | Value accepted - program continue as normal | - |
| 32 | input("Do you want your order to deliver in specific time? (Y/N): ") | Extreme valid data | nnnn | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 33 | input("Do you want your order to deliver in specific time? (Y/N): ") | Invalid data | q | Error massage: re-enter value | Error massage – program let user re-enter the value | Allow user to re-enter the correct choice |
| 34 | input("Do you want your order to deliver in specific time? (Y/N): ") | Extreme valid data | tttttt | Error massage: re-enter value | Error massage – program let user re-enter the value | Allow user to re-enter the correct choice |
| 35 | input("Do you want your order to deliver in specific time? (Y/N): ") | Erroneous data | 50 | Error massage: re-enter value | Error massage – program let user re-enter the value | Limit the input, allow user to input only y and n |
| 36 | input("Do you want your order to deliver in specific time? (Y/N): ") | Erroneous data | -\*32534534 | Error massage: re-enter value | Error massage – program let user re-enter the value | Limit the input, allow user to input only y and n |
| 37 | input("What time do you want your order to be deliver") | Typical valid data | 14:00 | Value accepted | Value accepted - program continue as normal | - |
| 38 | input("What time do you want your order to be deliver") | Extreme valid data | 180:00 | Error massage: re-enter value | Value accepted - program continue as normal | Limit range of number that can be inputted |
| 39 | input("What time do you want your order to be deliver") | Invalid data | e | Error massage: re-enter value | Value accepted - program continue as normal | Allow only number to be inputted |
| 40 | input("What time do you want your order to be deliver") | Extreme valid data | ttttttttttt | Error massage: re-enter value | Value accepted - program continue as normal | Allow only number to be inputted |
| 41 | input("What time do you want your order to be deliver") | Erroneous data | (1\*50) | Error massage: re-enter value | Value accepted - program continue as normal | Limit range of number that can be inputted |
| 42 | input("What time do you want your order to be deliver") | Erroneous data | print(“hi”) | Error massage: re-enter value | Value accepted - program continue as normal | Allow only number to be inputted |
| 43 | input("Confirm Information (Y/N):") | Typical valid data | Y | Value accepted | Value accepted - program continue as normal | - |
| 44 | input("Confirm Information (Y/N):") | Extreme valid data | nnnn | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 45 | input("Confirm Information (Y/N):") | Invalid data | q | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 46 | input("Confirm Information (Y/N):") | Extreme valid data | tttttt | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 47 | input("Confirm Information (Y/N):") | Erroneous data | 50 | Error massage: re-enter value | Error massage – program let user re-enter the value | Allow user to input only y and n |
| 48 | input("Confirm Information (Y/N):") | Erroneous data | -\*32534534 | Error massage: re-enter value | Error massage – program let user re-enter the value | Allow user to input only y and n |
| 49 | int(input("How many plate you want to order : ")) | Typical valid data | 1 | Value accepted | Value accepted - program continue as normal | - |
| 50 | int(input("How many plate you want to order : ")) | Extreme valid data | 100 | Value accepted | Value accepted - program continue as normal | - |
| 51 | int(input("How many plate you want to order : ")) | Invalid data | -1 | Error massage: re-enter value | program do not crash but the program doesn’t run as normal and no error massage | Limit range of number that can be inputted |
| 52 | int(input("How many plate you want to order : ")) | Extreme valid data | -789 | Error massage: re-enter value | program do not crash but the program doesn’t continue as normal and no error massage | Limit range of number that can be inputted |
| 53 | int(input("How many plate you want to order : ")) | Erroneous data | P | Error massage: re-enter value | Program crashes | Limit range of number that can be inputted |
| 54 | int(input("How many plate you want to order : ")) | Erroneous data | @$% | Error massage: re-enter value | Program crashes | Limit range of number that can be inputted |
| 55 | int(input("How many entress you want(minimum 2 maximum 5 entress): ")) | Typical valid data | 3 | Value accepted | Value accepted - program continue as normal | - |
| 56 | int(input("How many entress you want(minimum 2 maximum 5 entress): ")) | Extreme valid data | 70 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 57 | int(input("How many entress you want(minimum 2 maximum 5 entress): ")) | Invalid data | -3 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 58 | int(input("How many entress you want(minimum 2 maximum 5 entress): ")) | Extreme valid data | -70 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 59 | int(input("How many entress you want(minimum 2 maximum 5 entress): ")) | Erroneous data | y | Error massage: re-enter value | Program crashes | Error massage – program let user re-enter the value |
| 60 | int(input("How many entress you want(minimum 2 maximum 5 entress): ")) | Erroneous data | $#t | Error massage: re-enter value | Program crashes | Error massage – program let user re-enter the value |
| 61 | int(input("Please select  the entree: ")) | Typical valid data | 2 | Value accepted | Value accepted - program continue as normal | - |
| 62 | int(input("Please select  the entree: ")) | Extreme valid data | 20 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 63 | int(input("Please select  the entree: ")) | Invalid data | -2 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 64 | int(input("Please select  the entree: ")) | Extreme valid data | -20 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 65 | int(input("Please select  the entree: ")) | Erroneous data | r | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 66 | int(input("Please select  the entree: ")) | Erroneous data | %$EF | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 67 | int(input("Please select your side: ")) | Typical valid data | 2 | Value accepted | Value accepted - program continue as normal | - |
| 68 | int(input("Please select your side: ")) | Extreme valid data | 20 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 69 | int(input("Please select your side: ")) | Invalid data | -2 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 70 | int(input("Please select your side: ")) | Extreme valid data | -20 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 71 | int(input("Please select your side: ")) | Erroneous data | G | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 72 | int(input("Please select your side: ")) | Erroneous data | $%^ | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 73 | input("Confirm order (Y/N): ") | Typical valid data | Y | Value accepted | Value accepted - program continue as normal | - |
| 74 | input("Confirm order (Y/N): ") | Extreme valid data | y | Value accepted | Value accepted - program continue as normal | - |
| 75 | input("Confirm order (Y/N): ") | Invalid data | t | Error massage: re-enter value | Program respond like Y is inputted | Should let user input a correct data again |
| 76 | input("Confirm order (Y/N): ") | Extreme valid data | tttttttttttt | Error massage: re-enter value | Program respond like Y is inputted | Should let user input a correct data again |
| 77 | input("Confirm order (Y/N): ") | Erroneous data | 32 | Error massage: re-enter value | Program respond like Y is inputted | Should let user input a correct data again |
| 78 | input("Confirm order (Y/N): ") | Erroneous data | @#$%^ | Error massage: re-enter value | Program respond like Y is inputted | Should let user input a correct data again |
| 79 | input("Please insert discount code: ") | Typical valid data | happypanda | Value accepted | Value accepted - program continue as normal | - |
| 80 | input("Please insert discount code: ") | Extreme valid data | HaPpyPanDa | Value accepted | Value accepted - program continue as normal | - |
| 81 | input("Please insert discount code: ") | Invalid data | sadpanda | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 82 | input("Please insert discount code: ") | Extreme valid data | SaDPandAA | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 83 | input("Please insert discount code: ") | Erroneous data | -8541 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 84 | input("Please insert discount code: ") | Erroneous data | @#$%^& | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 85 | input("Do you want to re-enter discount code (Y/N): ") | Typical valid data | Y | Value accepted | Value accepted - program continue as normal | - |
| 86 | input("Do you want to re-enter discount code (Y/N): ") | Extreme valid data | y | Value accepted | Value accepted - error massage -program continue as normal | Error massage should not display |
| 87 | input("Do you want to re-enter discount code (Y/N): ") | Invalid data | t | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 88 | input("Do you want to re-enter discount code (Y/N): ") | Extreme valid data | tttttttttttt | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 89 | input("Do you want to re-enter discount code (Y/N): ") | Erroneous data | 32 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 90 | input("Do you want to re-enter discount code (Y/N): ") | Erroneous data | @#$%^ | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 91 | input("How you world like to pay(CASH/CARD): ") | Typical valid data | CASH | Value accepted | Value accepted - program continue as normal | - |
| 92 | input("How you world like to pay(CASH/CARD): ") | Extreme valid data | cash | Value accepted | Value accepted - program continue as normal | - |
| 93 | input("How you world like to pay(CASH/CARD): ") | Invalid data | No money | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 94 | input("How you world like to pay(CASH/CARD): ") | Extreme valid data | No MoNeyyyyy | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 95 | input("How you world like to pay(CASH/CARD): ") | Erroneous data | 852 | Error massage: re-enter value | Error massage – program let user re-enter the value | Allow only letter to be inputted |
| 96 | input("How you world like to pay(CASH/CARD): ") | Erroneous data | #$%^&\* | Error massage: re-enter value | Error massage – program let user re-enter the value | Allow only letter to be inputted |

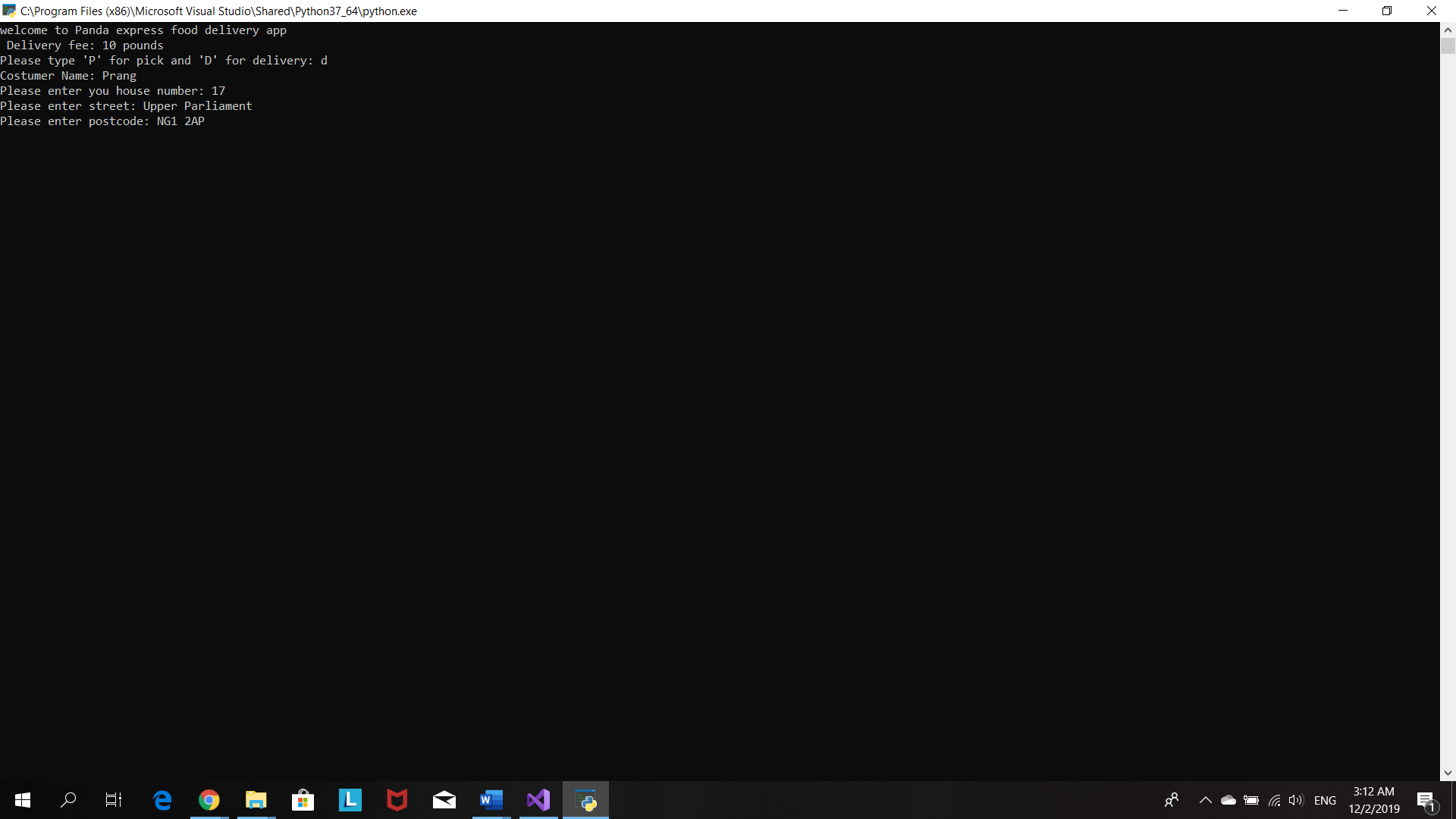
Screen shot

Start the program

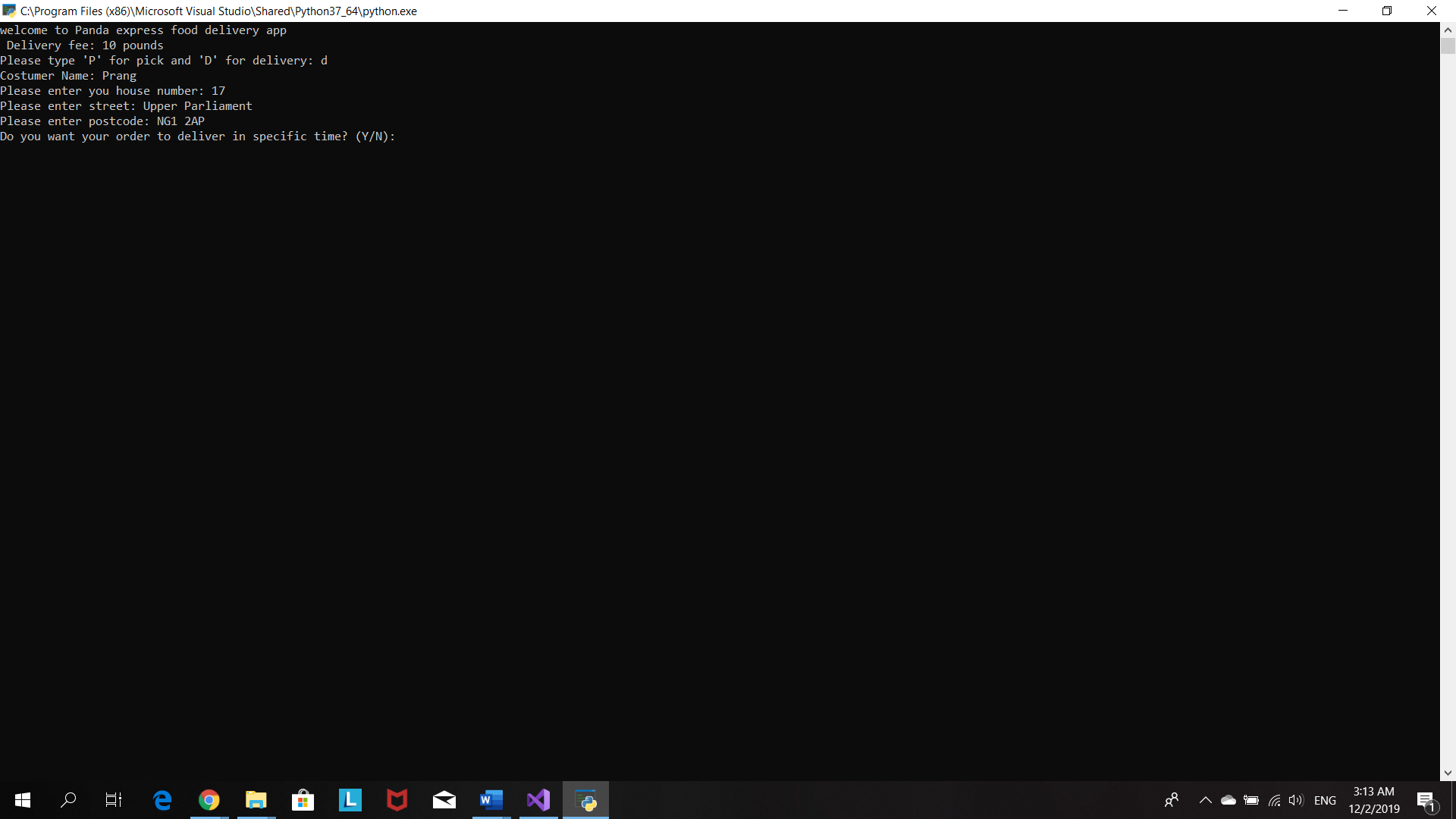
Ask costumer whether he/she wants order to be delivered or want to pick up the order by himself.



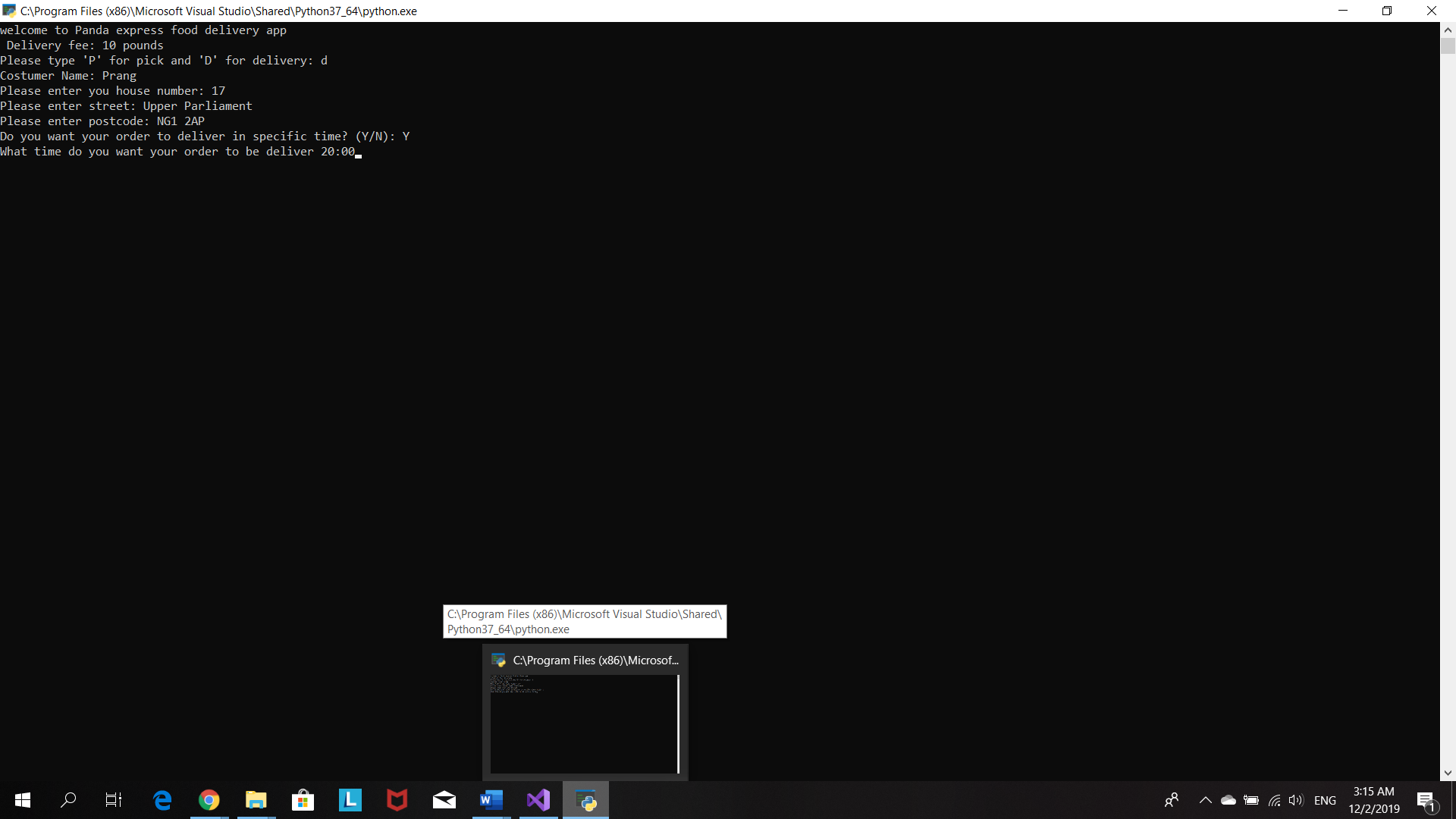
When costumer chooses ‘d’ for delivery , program will as for delivery details.



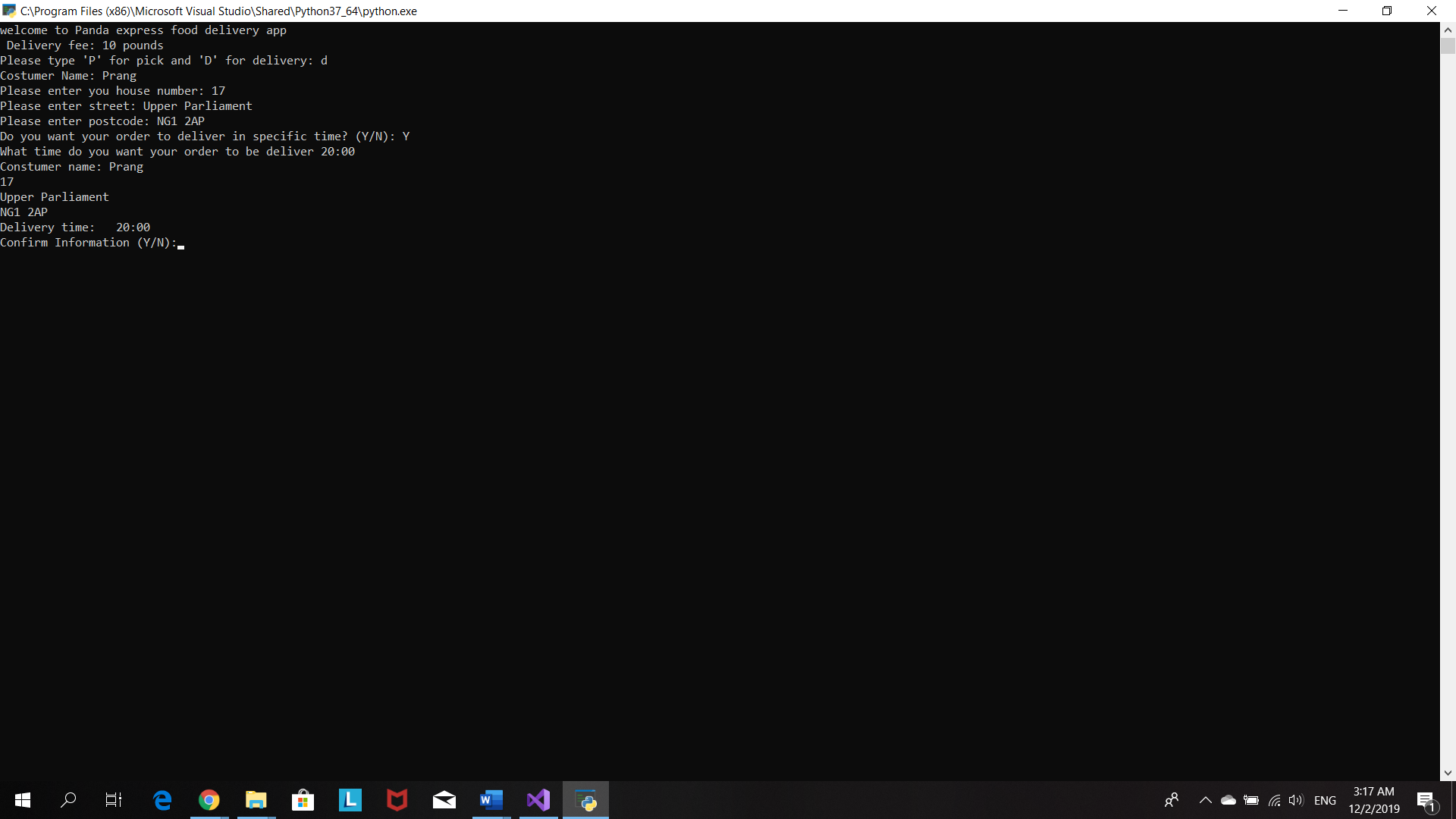
Program ask costumer do he/she wants to specific a delivery time.



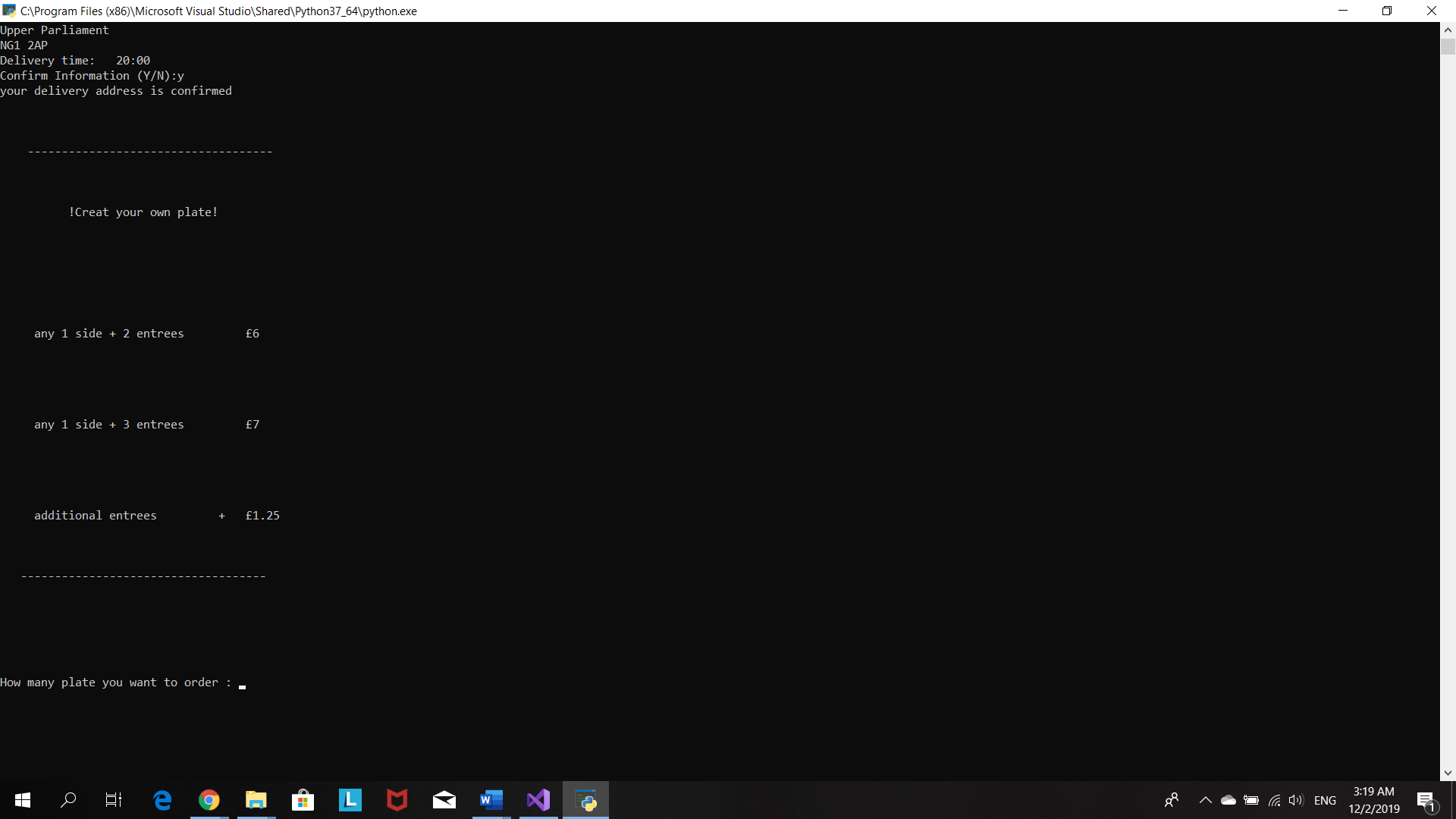
If costumer choose ‘y’ for yes, program will ask costumer to choose the time .



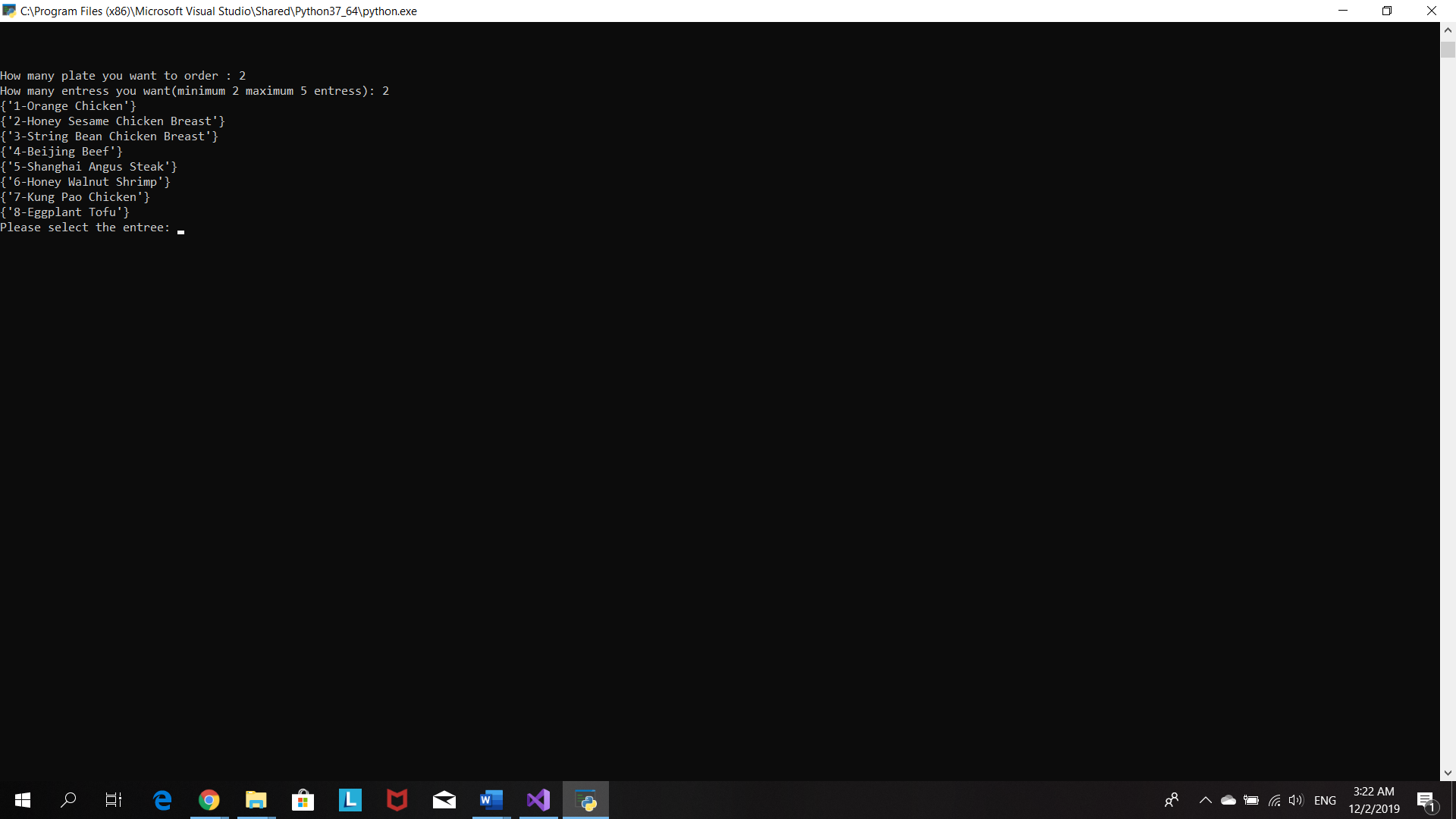
Program print out delivery details and ask costumer to confirm them.



Program ask costumer how many plate he/she wants to order



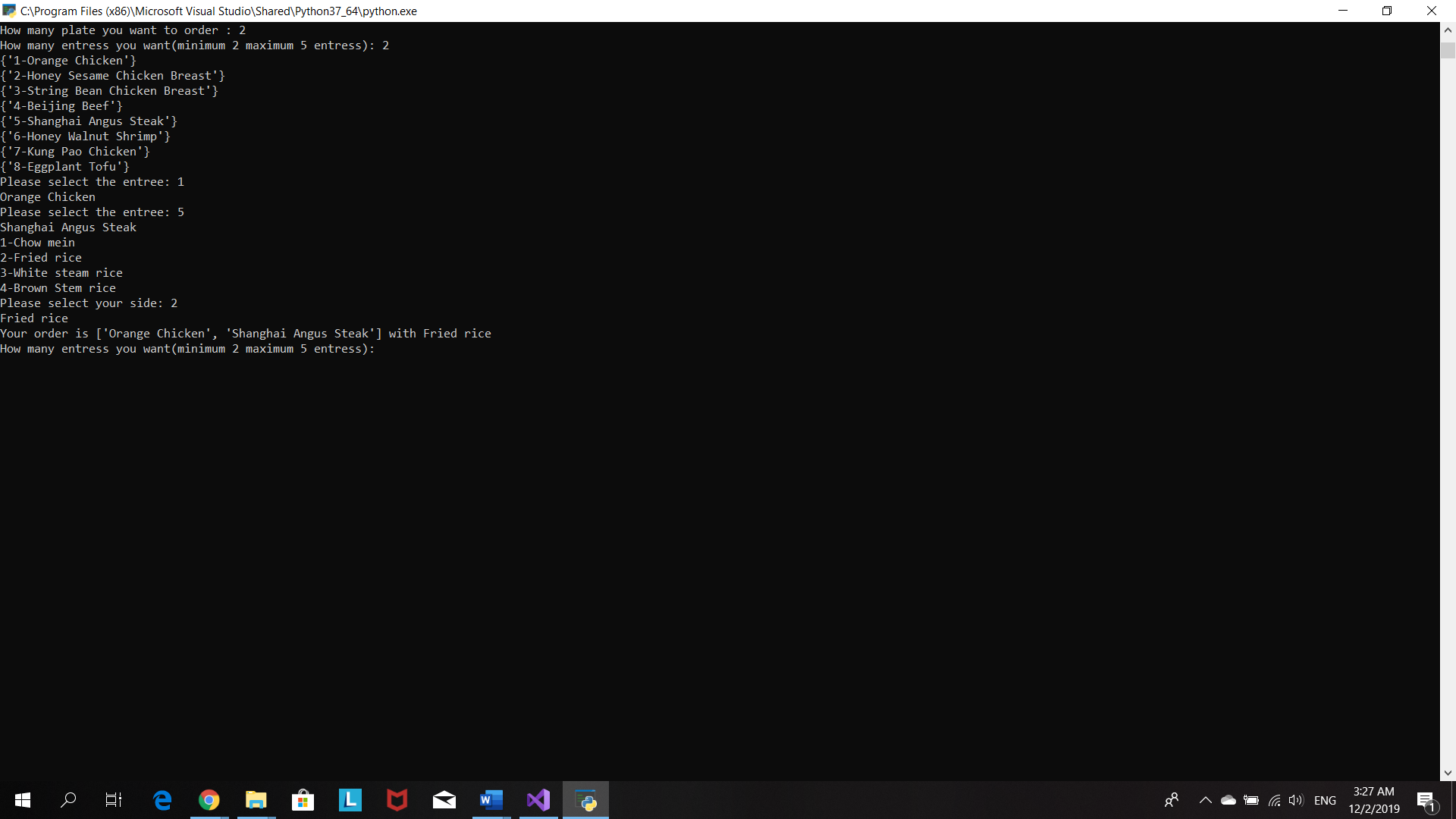
Then program ask costumer to how many entrée he/she wants and entrée menu display



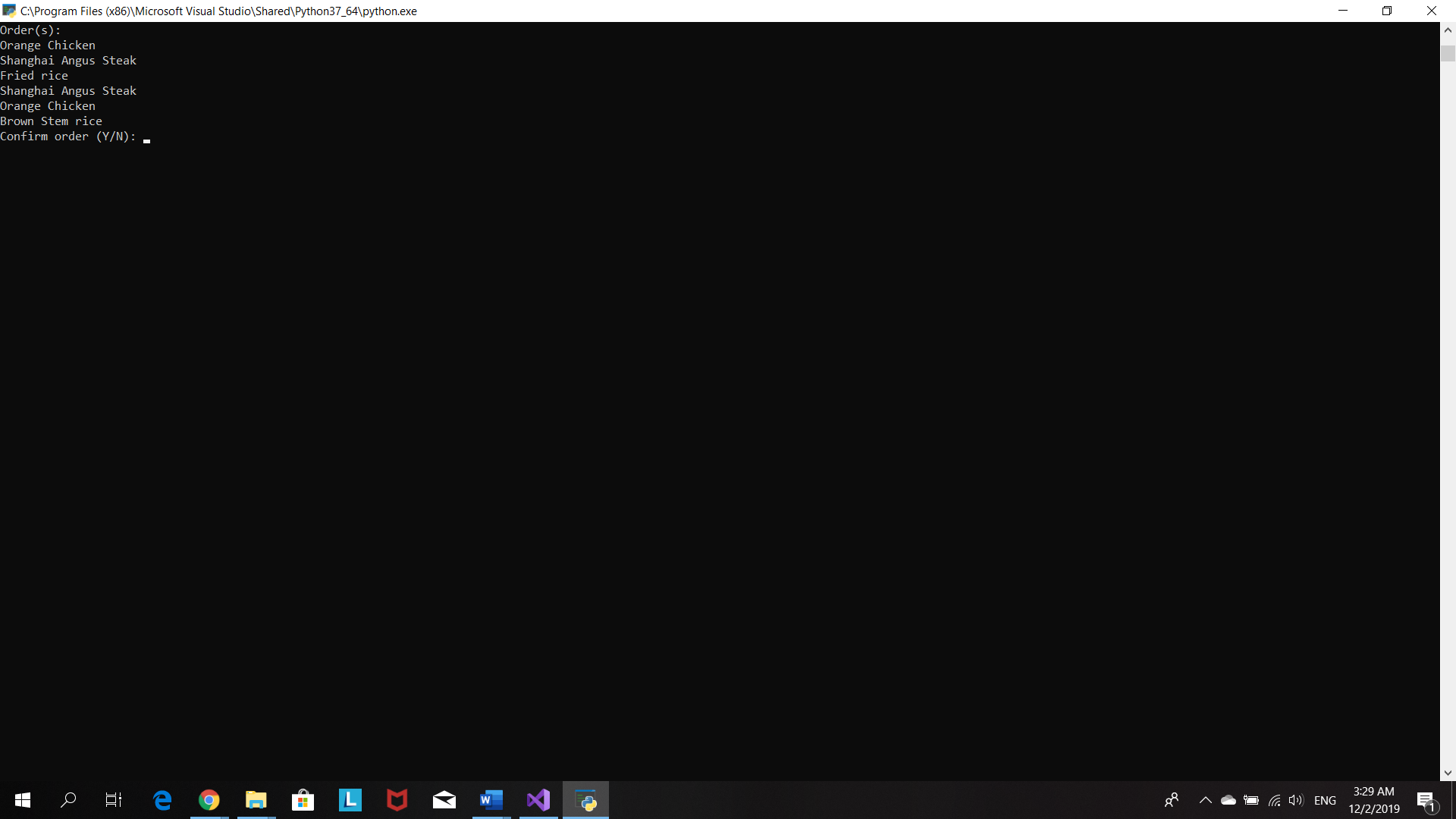
After costumer chooses entrees then program will display side menu and let costumer chooses



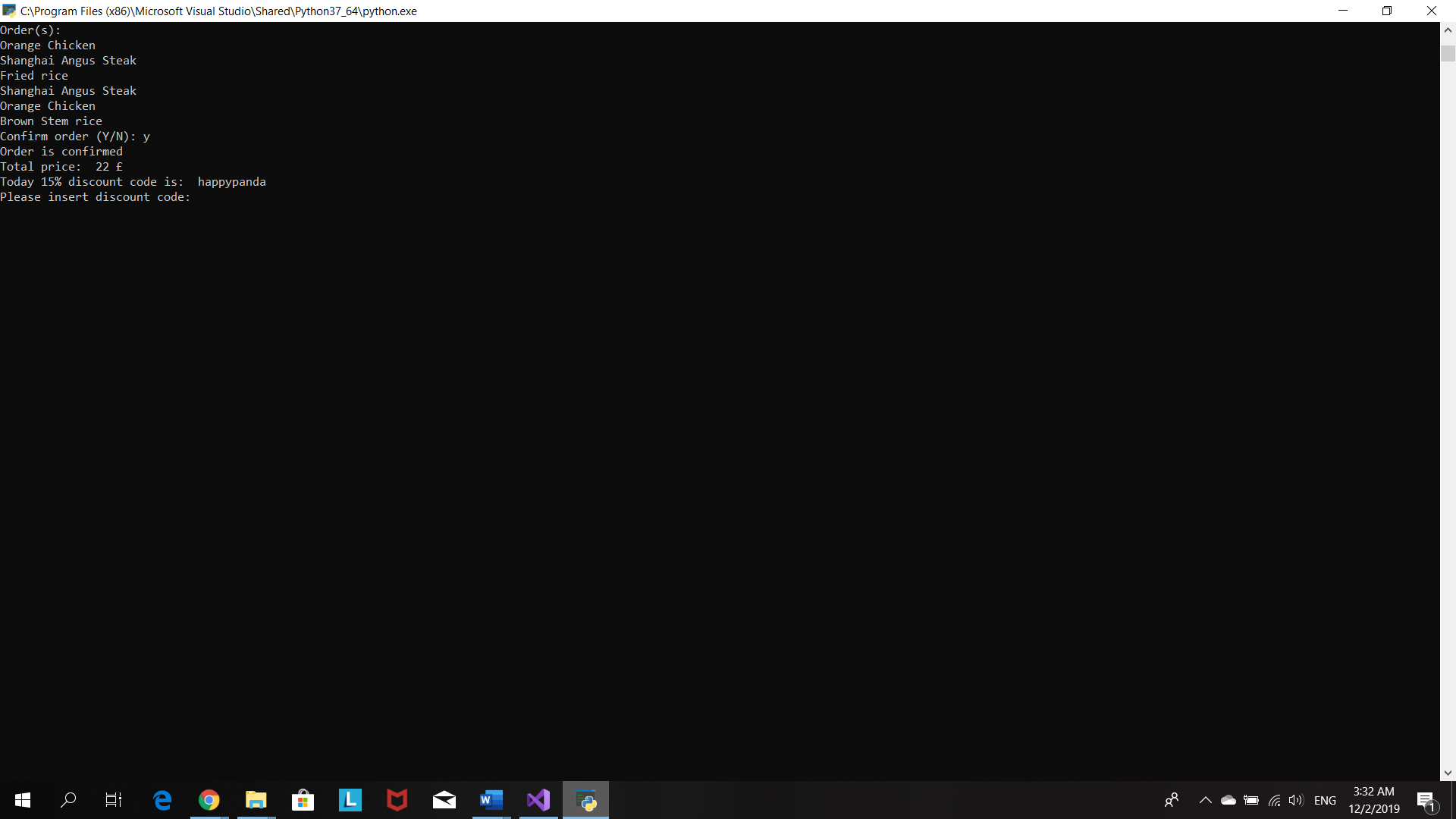
Costumer wants to order 2 plates so program let costumer chooses second plate



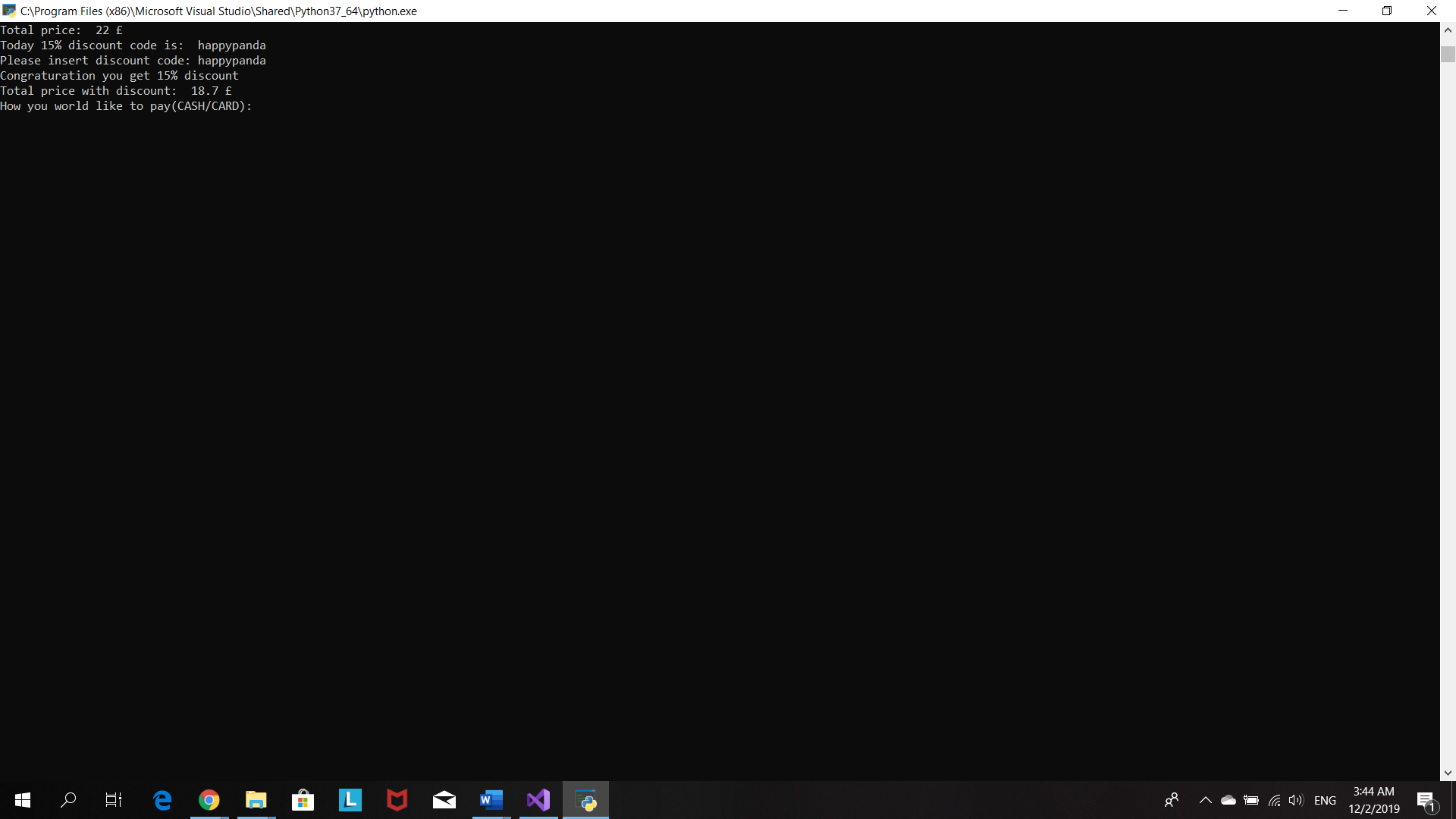
Program print order details and ask costumer to confirm them



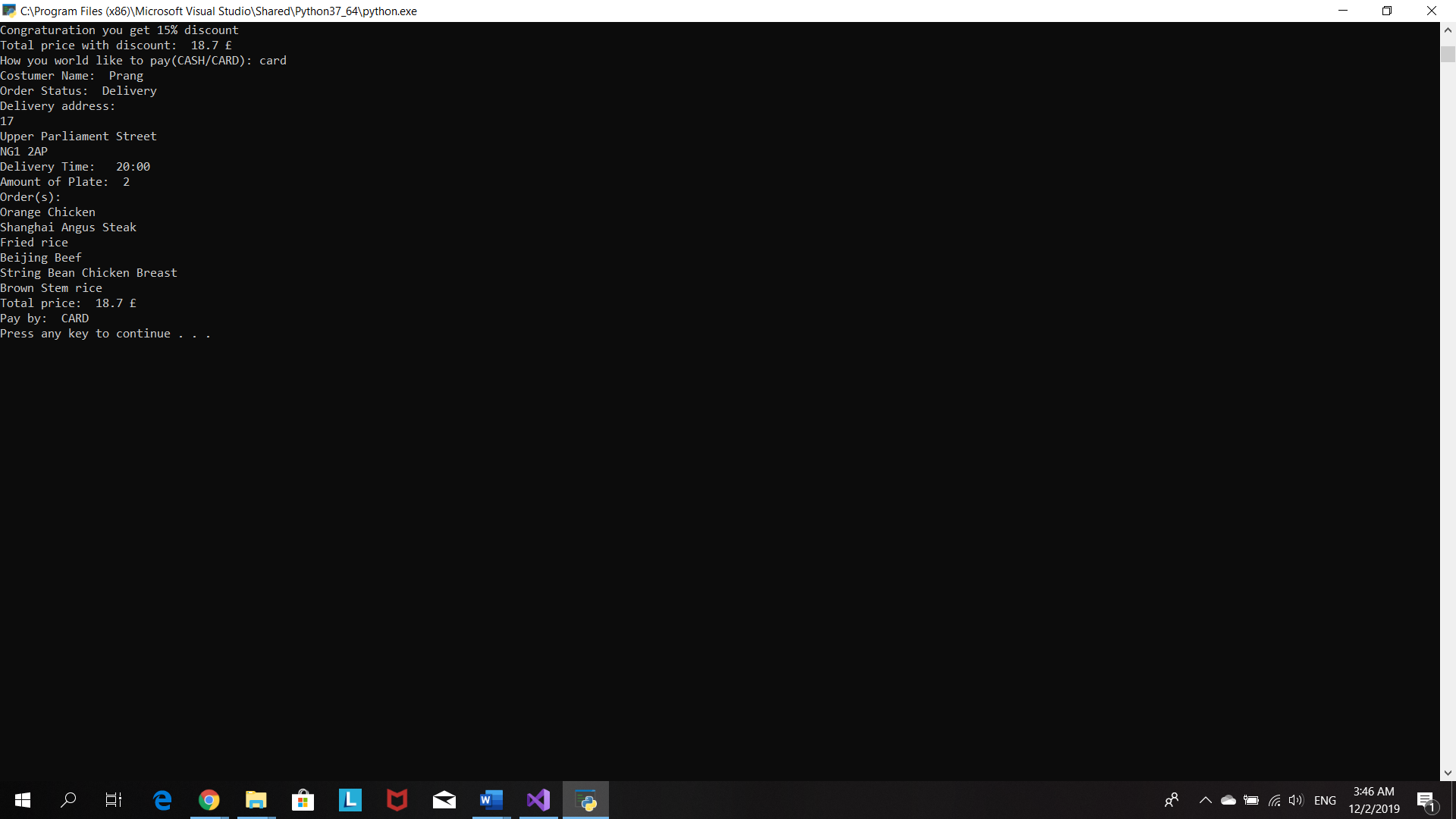
Program display a total price and inform costumer about the discount code



After costumer insert a correct code, program will re-calculate total price and ask how he/she wants to pay for order.



Program display all the details



# Source code

#Program 1: Food Delivery Program

#The purpose of the program is creat a food delivery app for local chinese resturant

#Name:Prang Kongthongluck

#Date:25/10/2019

#Version:1.0

#set up variable

price=()

totalPrice=()

payMethod=()

orderPrice=[]

userOrder=[]

menu=[

{"1-Orange Chicken"},

{"2-Honey Sesame Chicken Breast"},

{"3-String Bean Chicken Breast"},

{"4-Beijing Beef"},

{"5-Shanghai Angus Steak"},

{"6-Honey Walnut Shrimp"},

{"7-Kung Pao Chicken"},

{"8-Eggplant Tofu"}]

#printMenu is menu wittout number. It will be ues to print out the order for customer

printMenu=["Orange Chicken",

"Honey Sesame Chicken Breast",

"String Bean Chicken Breast",

"Beijing Beef",

"Shanghai Angus Steak",

"Honey Walnut Shrimp",

"Kung Pao Chicken",

"Eggplant Tofu"]

sideMenu=[

"1-Chow mein",

"2-Fried rice",

"3-White steam rice",

"4-Brown Stem rice"

]

#printsideMenu is sidemenu wittout number. It will be ues to print out the order for customer

printsideMenu=[

"Chow mein",

"Fried rice",

"White steam rice",

"Brown Stem rice"

]

userOrder=[]

#list locate inside for loop

printOrder=[]

address=[]

orderStatus=()

costumerName=()

deliveryTime=()

userentreeChoice=()

entree=()

amount=()

entreelist=[]

sidelist=[]

#---------------------------------------------------------------------------------------------------

print("welcome to Panda express food delivery app")

print("Today promotion: Free delivey  ")

#create a pickupDelivery function

def pickupDelivery():

#set some valiables as globle so it can be use outside their function

global customerName

global orderStatus

global deliveryTime

#ask customer what they prefer pick up or delivery

delivery=input("Please type 'P' for pick and 'D' for delivery: ")

#change input from delivery to be uppercase

delivery=delivery.upper()

#ask for customer name

customerName=input("Costumer Name: ")

#make customerName begin with capital letter

customerName=customerName.capitalize()

#when customer choose delivery option

if delivery== "D":

orderStatus='Delivery'

#loop to confirm deliver address

while True:

houseNumber=input("Please enter you house number: ")

# add houseNumber in to addres list

address.append(houseNumber)

street=input("Please enter street: ")

# add street in to addres list

address.append(street)

postcode=input("Please enter postcode: ")

# add postcode in to addres list

address.append(postcode)

deliveryTimeAsk=input("Do you want your order to deliver in specific time? (Type Y for yes and N for no): ")#make input in deliveryTimeAsk to be uppercase

delivertTimeAsk=deliveryTimeAsk.upper()

#if customer what their order to deliver in specific time

if (deliveryTimeAsk=="Y"):

deliveryTime=input("What time do you want your order to be deliver")#if customer dont what their order to deliver in specific time. Their order will be delivered as soon as possible

else:

deliveryTime="ASAP"

print("Your order wil be deliver as soon as possible")

#print out delivery details

print("Customer name:",costumerName)

#print address list vertically

for a in (address):

print (a)

print("Delivery time: ",deliveryTime)

#make customer to re checkt their delivery details

confirmAddress=input("Confirm Information (Type Y for yes and N for no):")

#make input of confirmAddress alwasy uppercase

confirmAddress=confirmAddress.upper()

#costumer dont confirm delivery details

if confirmAddress=="N":

#remove the data in the address list when constumer dont confirm the divery adress

address.remove(houseNumber)

address.remove(street)

address.remove(postcode)

print("Your delivery address has been cancel")

print("Please enter your delivery adress")

#customer confirm delivery details

elif confirmAddress=="Y":

print("your delivery address is confirmed")

break

else:

print("wrong input. Please try again")

#customer choose to pick up order themselves

if delivery=="P":

orderStatus='Pick Up'

print("Customer Name: ",customerName)

print("Customer will pick up the order")

#call pickupDelivery function

pickupDelivery()

print(

"""

    ------------------------------------

          !Creat your own plate!

     any 1 side + 2 entrees         £6

     any 1 side + 3 entrees         £7

     additional entrees         +   £1.25

   ------------------------------------

    """

)

#create order function

def order():

#set some valiables as globle so it can be use outside their function

global entree

global amount

#the purpose of this loop is to confirm order details

while True:

amount=int(input("How many plate you want to order : "))

#make a loop repeat respectively to amount of plate costumer choosed

for i in range(amount):

#the purpose of this loop is to make sure number of entree costumer choose is between 2 and 5

while True:

entree=int(input("How many entress you want(minimum 2 entrees and maximum 5 entress): "))

#make sure number of entree costumer choose is between 2 and 5

if (entree<=1) or (entree>=5):

print("INCORRECT INPUT.Please try again")

else:

break

#this will help to calculate the price

if (entree==2):

price=6

if (entree==3):

price=7

if (entree==4):

price=8.25

if (entree==5):

price=9.5

#add price in orderPrice list

orderPrice.append(price)

#display menu vertically

for i in (menu):

print (i)

#Purpose of this list is to display the order for costumer inside the loop so it can print each order seperatly

printOrder=[]

#make a loop repeat respectively to number of entree costumer choose

for i in range(entree):

#make sure user input right number od entree(1-8)

while True:

entreeChoice=int(input("Please selet the entree: "))

if (entreeChoice<1) or (entreeChoice>8):

print("INVALID INOUT.Please choose number 1-8")

else:

break

#to print out the correct entree choice of costumer

userentreeChoice=printMenu[entreeChoice-1]

print(userentreeChoice)

#insert user's choice of entrees in userOrder list and entreelist

userOrder.append(userentreeChoice)

entreelist.append(userentreeChoice)

#use for loop to print side menu vertically

for x in sideMenu:

print (x)

sideChoice=int(input("Please select your side: "))

#to print out the correct side dish choice of costumer

userSideChoice=printsideMenu[sideChoice-1]

print(userSideChoice)

#insert user's choice of side dish to userOrder list and sidelist

userOrder.append(userSideChoice)

sidelist.append(userSideChoice)

#print out customer choice

def OrderRecipe(choiceE,choiceS):

print("Your order is",choiceE,"with",choiceS)

OrderRecipe(entreelist,sidelist)

#customer order morethen 1 plate

#data in printOrder list will be deleted so that program able to print out each plate choic individually

if (amount>=2):

entreelist.remove(userentreeChoice)

sidelist.remove(userSideChoice)

#display order details for customer to confirm

print("number of plate: ",amount)

print("Order(s): ")

#print order details vertically

for i in (userOrder):

print (i)

confirmOrder=input("Confirm order (Type Y for yes and N for no): ")

confirmOrder.upper()

#allow customer to confirm their order

if (confirmOrder=="N"):

print("Order has been cancel")

print("Please order again")

else:

print("Order is confirmed")

break

#call order function

order()

#create payment function

def payment():

#set some valiables as globle so it can be use outside their function

global totalPrice

global payMethod

#total price is sum of number in orderPrice list

totalPrice=sum(orderPrice)

print("Total price: ",totalPrice,"£")

#inform customer discount code

print("Today 15% discount code is:  happypanda")

#make a loop TRUE

reenter="Y"

#the purpose of this loop is to make sure customer enter the correct discount code

while (reenter=="Y"):

discount=input("Please insert discount code: ")

#make input from discount always be in lower case

discount=discount.lower()

if (discount == "happypanda"):

#totelprice reduce by 15%

totalPrice=totalPrice-(0.15\*totalPrice)

print("Congraturation you get 15% discount")

print("Total price with discount: ",totalPrice,"£")

break

else:

print("Incorret discount code")

#ask customer do they want to re-enter the code

reenter=input("Do you want to re-enter discount code (Type Y for yes and N for no): ")

reenter=reenter.upper()

if (reenter == "N"):

break

else:

print("Incorrect input")

#Ask customer how they would like to pay

payMethod=input("How you world like to pay(CASH/CARD): ")

payMethod=payMethod.upper()

#check whether customer select the collect payMethod

while True:

if (payMethod=="CASH") or (payMethod=="CARD"):

break

else:

print("Incorrect in out. Please select payment method")

#call payment function

payment()

#create receipt function

#allow customer to see all details(delivery,order and payment)

def receipt():

print ("Customer Name: ",customerName)

print("Order Status: ",orderStatus)

if (orderStatus=="Delivery"):

print("Delivery address: ")

for i in address:

print (i)

print("Delivery Time: ",deliveryTime)

print("Amount of Plate: ",amount)

print("Order(s): ")

for i in userOrder:

print(i)

print("Total price: ",totalPrice,"£")

print("Pay by: ",payMethod)

#call receipt function

# receipt()

Reference

101 Computing. (2017). *Italian Takeaway Ordering system*. [online] Available at: https://www.101computing.net/italian-takeaway-ordering-system/

GeeksforGeeks. (2019). *Print lists in Python (4 Different Ways) - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/print-lists-in-python-4-different-ways/

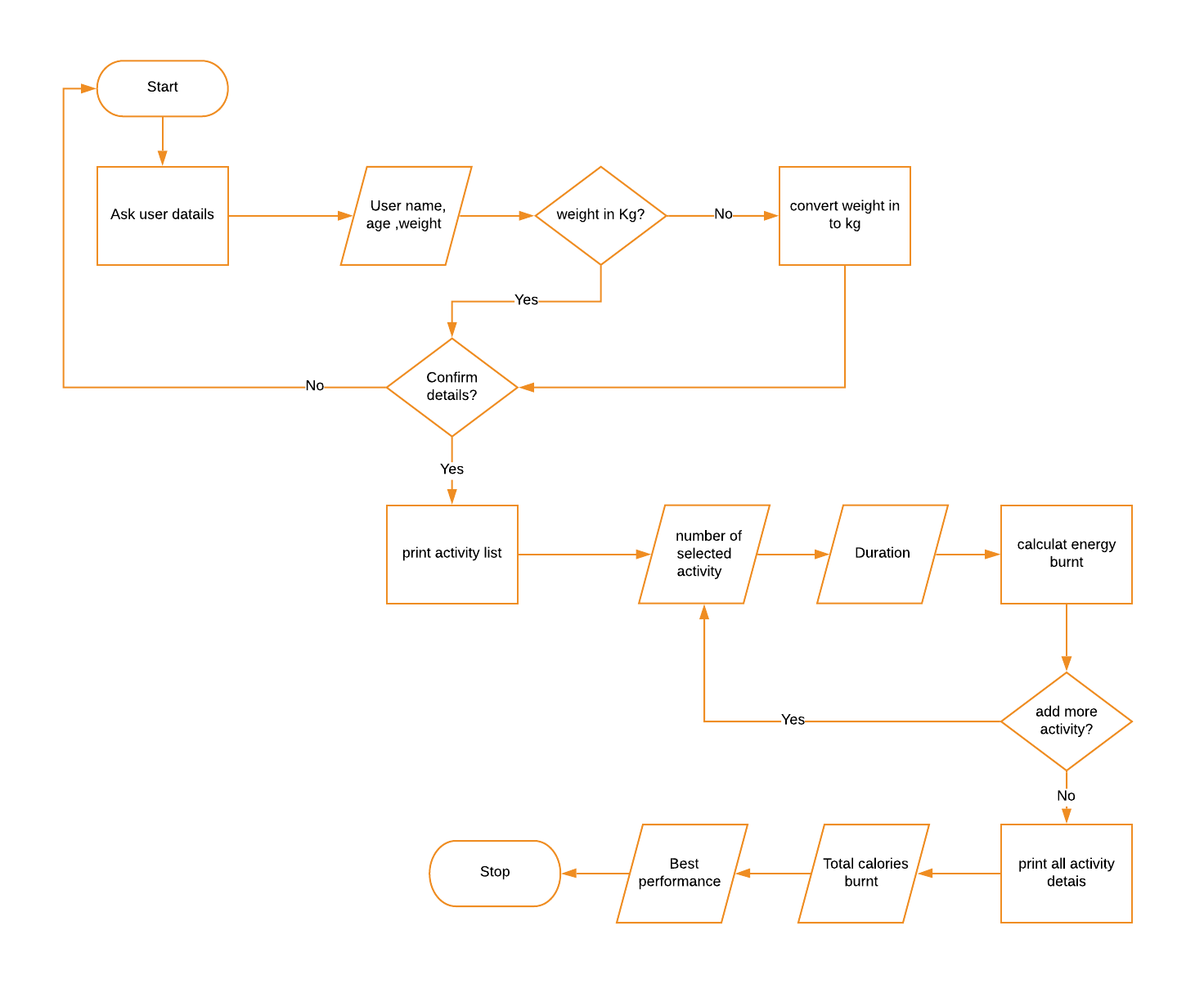
GeeksforGeeks. (2019). *Python | Remove all characters except letters and numbers - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/python-remove-all-characters-except-letters-and-numbers/

Harrington, A. (2019). *1.11. Defining Functions of your Own — Hands-on Python Tutorial for Python 3*. [online] Anh.cs.luc.edu. Available at: http://anh.cs.luc.edu/python/hands-on/3.1/handsonHtml/functions.html

Programiz.com. (2019). *Python List remove()*. [online] Available at: https://www.programiz.com/python-programming/methods/list/remove [Accessed 1 Dec. 2019].

Program 2: Calories Burnt Program Report

# Calories burnt program flow chart



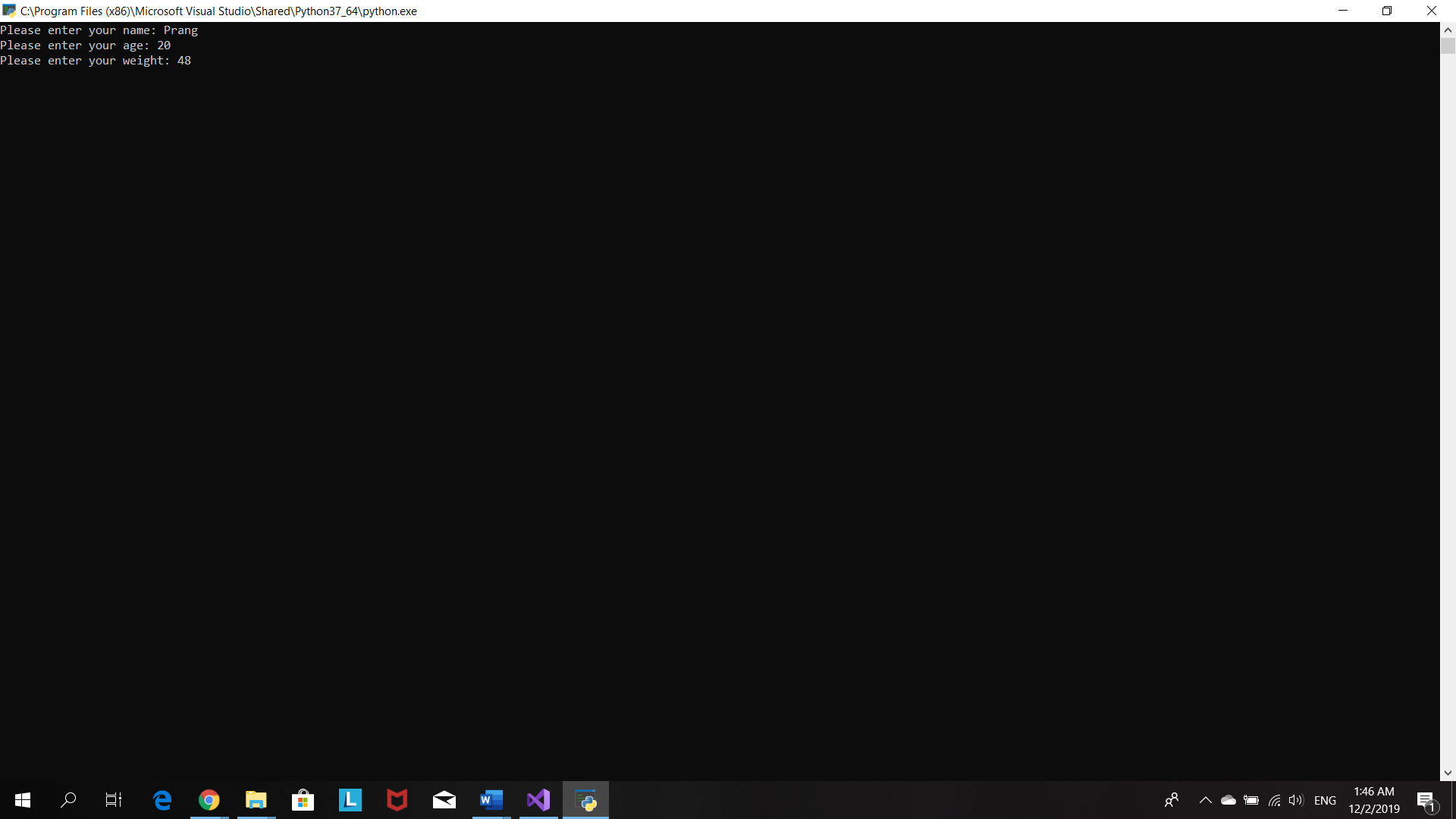
# Testing Table- Program 2: Calories Burnt Program

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. | Item to test | Test description | Test input | Expected result | Actual result | Comment |
| 1 | input("Please enter your name: ") | Typical valid data | Jay | Value accepted | Value accepted-program continue as normal | - |
| 2 | input("Please enter your name: ") | Extreme valid data | pppppppppppppppppp | Value accepted | Value accepted-program continue as normal | - |
| 3 | input("Please enter your name: ") | Invalid data | 10 | Error massage: re-enter value | Value accepted-program continue as normal | Allow only letter to be inputted |
| 4 | input("Please enter your name: ") | Invalid extreme data | -810 | Error massage: re-enter value | Value accepted-program continue as normal | Allow only letter to be inputted |
| 5 | input("Please enter your name: ") | Erroneous data | $%^&(() | Error massage: re-enter value | Value accepted-program continue as normal | Allow only letter to be inputted |
| 6 | input("Please enter your name: ") | Erroneous data | “HELP” | Error massage: | Value accepted-program continue as normal | Allow only letter to be inputted |
| 7 | input("Please enter your age: ") | Typical valid data | 18 | Value accepted | Value accepted-program continue as normal | - |
| 8 | input("Please enter your age: ") | Extreme valid data | 200 | Error massage: re-enter value | Value accepted-program continue as normal | Set the limit of the input number |
| 9 | input("Please enter your age: ") | Invalid data | -18 | Error massage: re-enter value | Value accepted-program continue as normal | Set the limit of the input number |
| 10 | input("Please enter your age: ") | Invalid extreme data | -200 | Error massage: re-enter value | Value accepted-program continue as normal | Set the limit of the input number |
| 11 | input("Please enter your age: ") | Erroneous data | Hi | Error massage: re-enter value | Value accepted-program continue as normal | Allow only number to be inputted |
| 12 | input("Please enter your age: ") | Erroneous data | Honorificabilitudinitatibus!! | Error massage: re-enter value | Value accepted-program continue as normal | Allow only number to be inputted |
| 13 | int(input("Please enter your weight: ")) | Typical valid data | 60 | Value accepted | Value accepted-program continue as normal | - |
| 14 | int(input("Please enter your weight: ")) | Extreme valid data | 20000 | Error massage: re-enter value | Value accepted-program continue as normal | Limit the range of inputted number |
| 15 | int(input("Please enter your weight: ")) | Invalid data | -60 | Error massage: re-enter value | Value accepted-program continue as normal | Limit the range of inputted number |
| 16 | int(input("Please enter your weight: ")) | Invalid extreme data | -20000 | Error massage: re-enter value | Value accepted-program continue as normal | Limit the range of inputted number |
| 17 | int(input("Please enter your weight: ")) | Erroneous data | Hi | Error massage: re-enter value | Error massage – user re-enter value | - |
| 18 | int(input("Please enter your weight: ")) | Erroneous data | Honorificabilitudinitatibus!! | Error massage: re-enter value | Error massage – user re-enter value | - |
| 19 | input("Please choose unite of your weight you entered (P=pound/KG): ") | Typical valid data | P | Value accepted | Value accepted-program continue as normal | - |
| 20 | input("Please choose unite of your weight you entered (P=pound/KG): ") | Extreme valid data | p | Value accepted | Value accepted-program continue as normal | - |
| 21 | input("Please choose unite of your weight you entered (P=pound/KG): ") | Invalid data | r | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 22 | input("Please choose unite of your weight you entered (P=pound/KG): ") | Invalid extreme data | rrrrrrr | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 23 | input("Please choose unite of your weight you entered (P=pound/KG): ") | Erroneous data | 852 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 24 | input("Please choose unite of your weight you entered (P=pound/KG): ") | Erroneous data | $%^&\* | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 25 | input("Confirm your information (Y/N)") | Typical valid data | Y | Value accepted | Value accepted-program continue as normal | - |
| 26 | input("Confirm your information (Y/N)") | Extreme valid data | y | Value accepted | Value accepted-program continue as normal | - |
| 27 | input("Confirm your information (Y/N)") | Invalid data | t | Error massage: re-enter value | No error massages. Program respond like “N” is inputted | Error massage – program let user re-enter the value |
| 28 | input("Confirm your information (Y/N)") | Invalid extreme data | tttttttttttt | Error massage: re-enter value | No error massages. Program respond like “N” is inputted | Error massage – program let user re-enter the value |
| 29 | input("Confirm your information (Y/N)") | Erroneous data | 32 | Error massage: re-enter value | No error massages. Program respond like “N” is inputted | Error massage – program let user re-enter the value |
| 30 | input("Confirm your information (Y/N)") | Erroneous data | @#$%^ | Error massage: re-enter value | No error massages. Program respond like “N” is inputted | Error massage – program let user re-enter the value |
| 31 | open ("mettable.txt") | File is renamed |  | Error massage,  open back-up file | Error massage – program continues without data in file | Create same file using same name as a back-up file but store in different places |
| 32 | open ("mettable.txt") | File is moved |  | Error massage,  open back-up file | Error massage – program continues without data in file | Create same file using same name as a back-up file but store in different places |
| 33 | open ("mettable.txt") | File is deleted |  | Error massage,  open back-up file | Error massage – program continues without data in file | Create same file using same name as a back-up file but store in different places |
| 34 | open ("mettable.txt") | File doesn’t exit |  | Error massage,  open back-up file | Error massage – program continues without data in file | Create same file using same name as a back-up file but store in different places |
| 35 | int(input("Please select your activity by input number located in front of met and activity name: ")) | Typical valid data | 1 | Value accepted | Value accepted-program continue as normal | - |
| 36 | int(input("Please select your activity by input number located in front of met and activity name: ")) | Extreme valid data | 100 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 37 | int(input("Please select your activity by input number located in front of met and activity name: ")) | Invalid data | -1 | Error massage: re-enter value | Error massage - program let user re-enter the value | - |
| 38 | int(input("Please select your activity by input number located in front of met and activity name: ")) | Invalid extreme data | -100000 | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 39 | int(input("Please select your activity by input number located in front of met and activity name: ")) | Erroneous data | h | Error massage: re-enter value | Error massage - program let user re-enter the value | - |
| 40 | int(input("Please select your activity by input number located in front of met and activity name: ")) | Erroneous data | #$%^& | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 41 | int(input("How long you do the activity(in minute): ")) | Typical valid data | 45 | Value accepted | Value accepted-program continue as normal | - |
| 42 | int(input("How long you do the activity(in minute): ")) | Extreme valid data | 1000000 | Error massage: re-enter value | Value accepted-program continue as normal | Limit range of number |
| 43 | int(input("How long you do the activity(in minute): ")) | Invalid data | -45 | Error massage: re-enter value | Value accepted-program continue as normal | Limit range of number |
| 44 | int(input("How long you do the activity(in minute): ")) | Invalid extreme data | -78924 | Error massage: re-enter value | Value accepted-program continue as normal | Limit range of number |
| 45 | int(input("How long you do the activity(in minute): ")) | Erroneous data | Float(100.00) | Error massage: re-enter value | Error massage - program let user re-enter the value | - |
| 46 | int(input("How long you do the activity(in minute): ")) | Erroneous data | #$%^&\*( | Error massage: re-enter value | Error massage – program let user re-enter the value | - |
| 47 | input("Do you want to add more activities(Y/N): ") | Typical valid data | Y | Value accepted | Value accepted-program continue as normal | - |
| 48 | input("Do you want to add more activities(Y/N): ") | Extreme valid data | n | Value accepted | Value accepted-program continue as normal | - |
| 49 | input("Do you want to add more activities(Y/N): ") | Invalid data | Pp | Error massage: re-enter value | Error massage – program doesn’t let user to re-enter correct value but continue as “N” is inputted | Let user re-enter correct value |
| 50 | input("Do you want to add more activities(Y/N): ") | Invalid extreme data | Rtyuiknbvcdrftyuio | Error massage: re-enter value | Error massage – program doesn’t let user to re-enter correct value but continue as “N” is inputted | Let user re-enter correct value |
| 51 | input("Do you want to add more activities(Y/N): ") | Erroneous data | @#$%^&\* | Error massage: re-enter value | Error massage – program doesn’t let user to re-enter correct value but continue as “N” is inputted | Let user re-enter correct value |
| 52 | input("Do you want to add more activities(Y/N): ") | Erroneous data | Print(“Hello”) | Error massage: re-enter value | Error massage – program doesn’t let user to re-enter correct value but continue as “N” is inputted | Let user re-enter correct value |

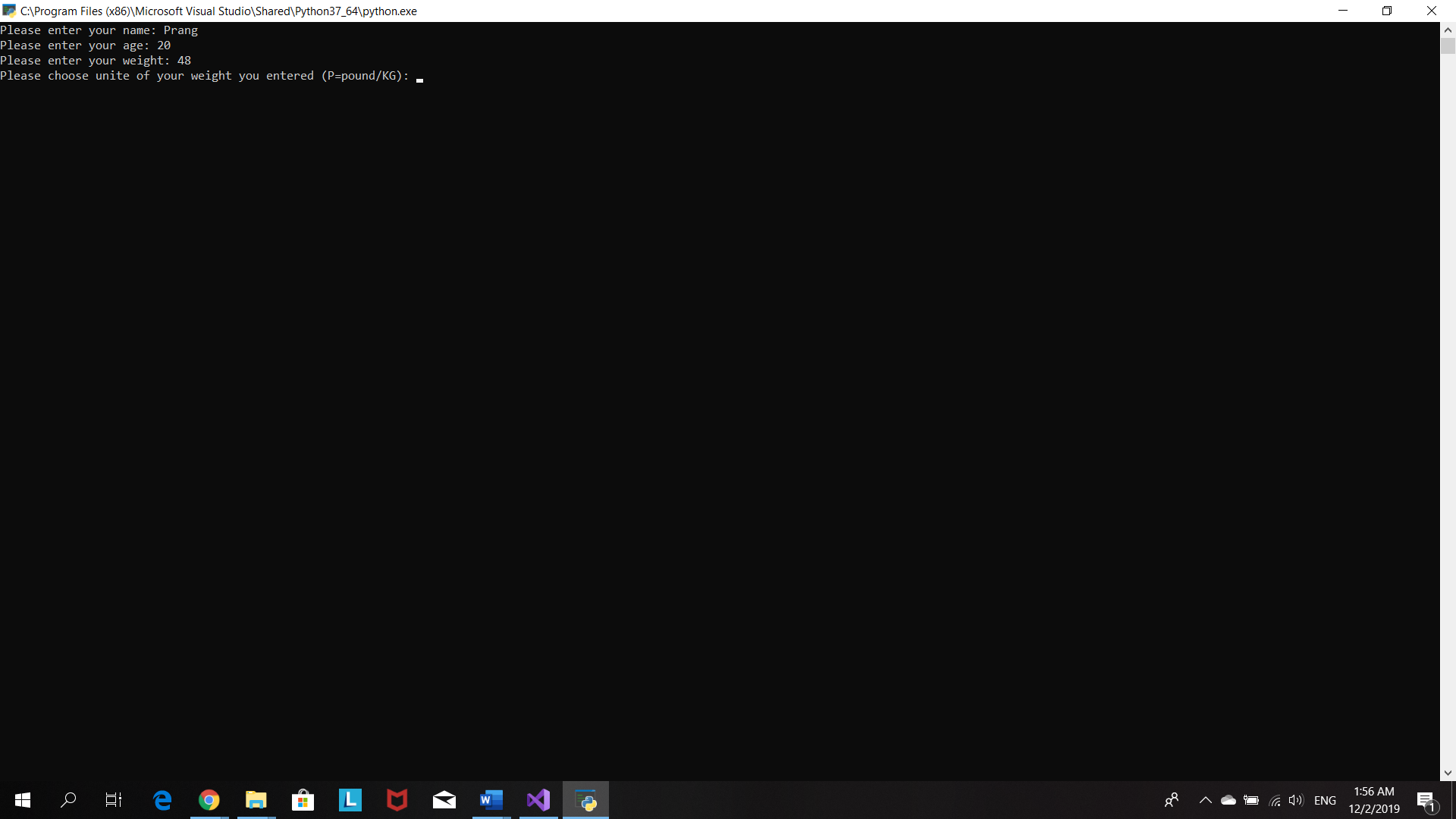
Screen shot

Start the program

Ask user details

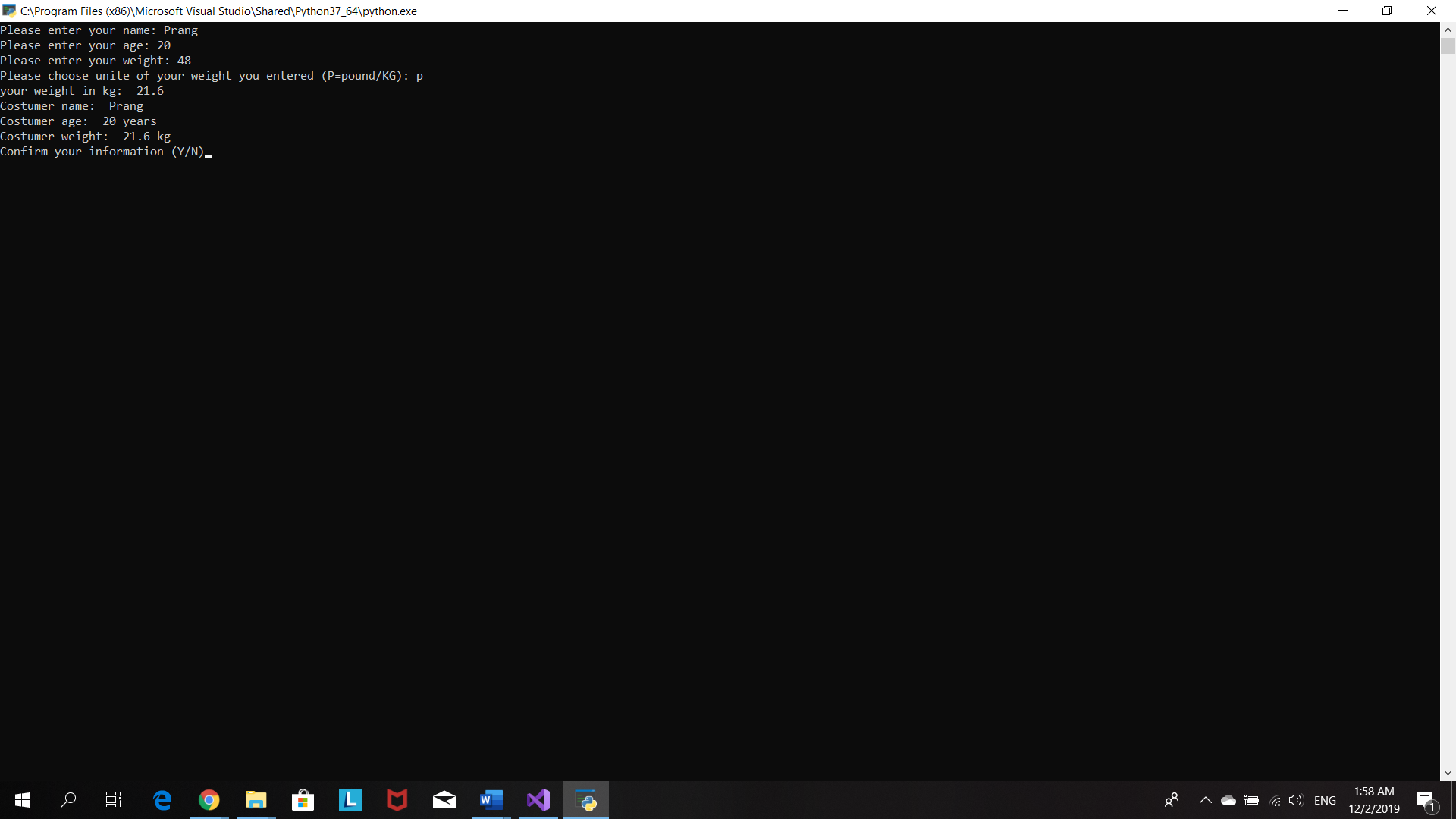


User chooses weight unit

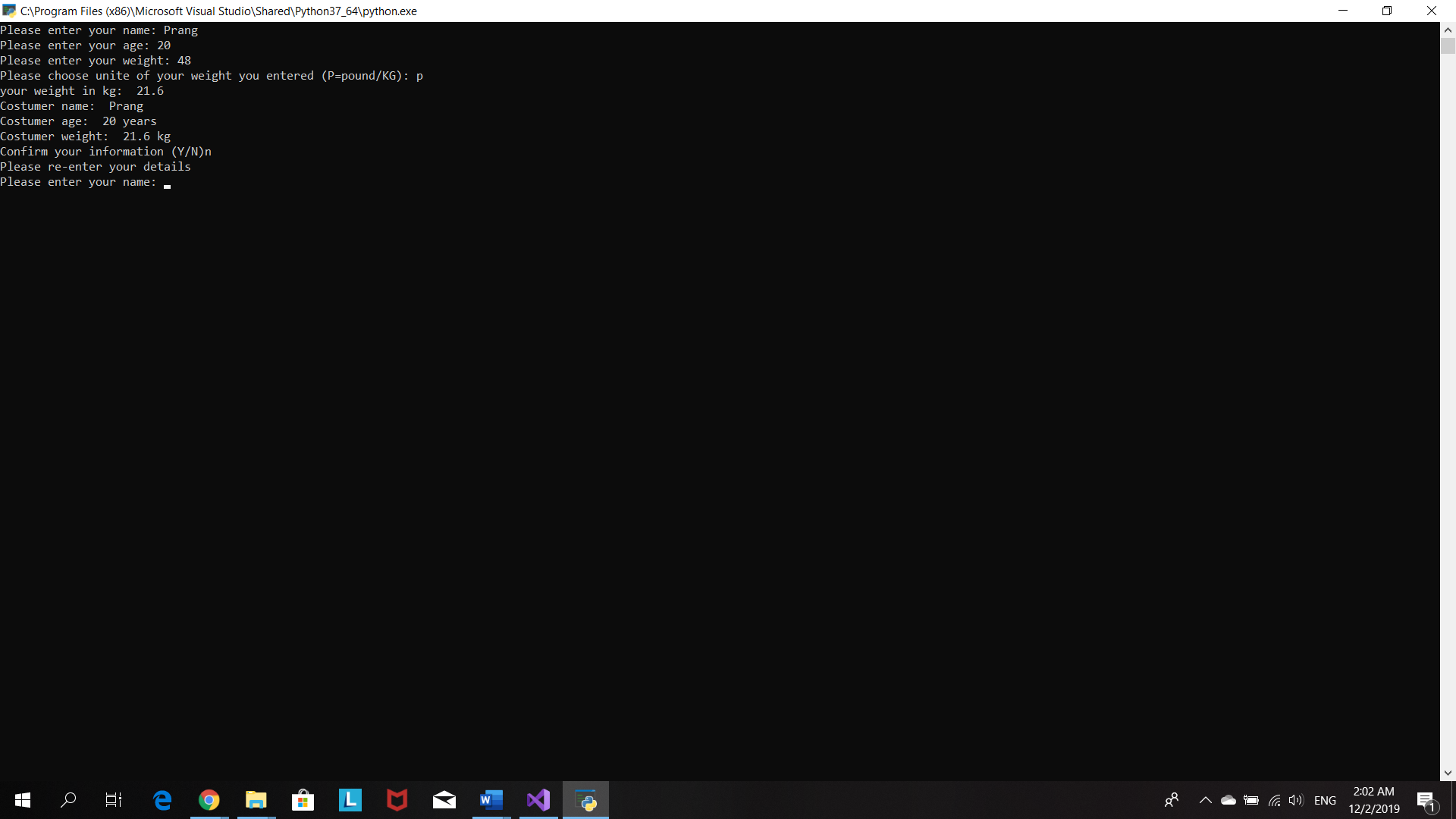


If user chooses ‘p’ for pound program will convert the input weight in to kg .

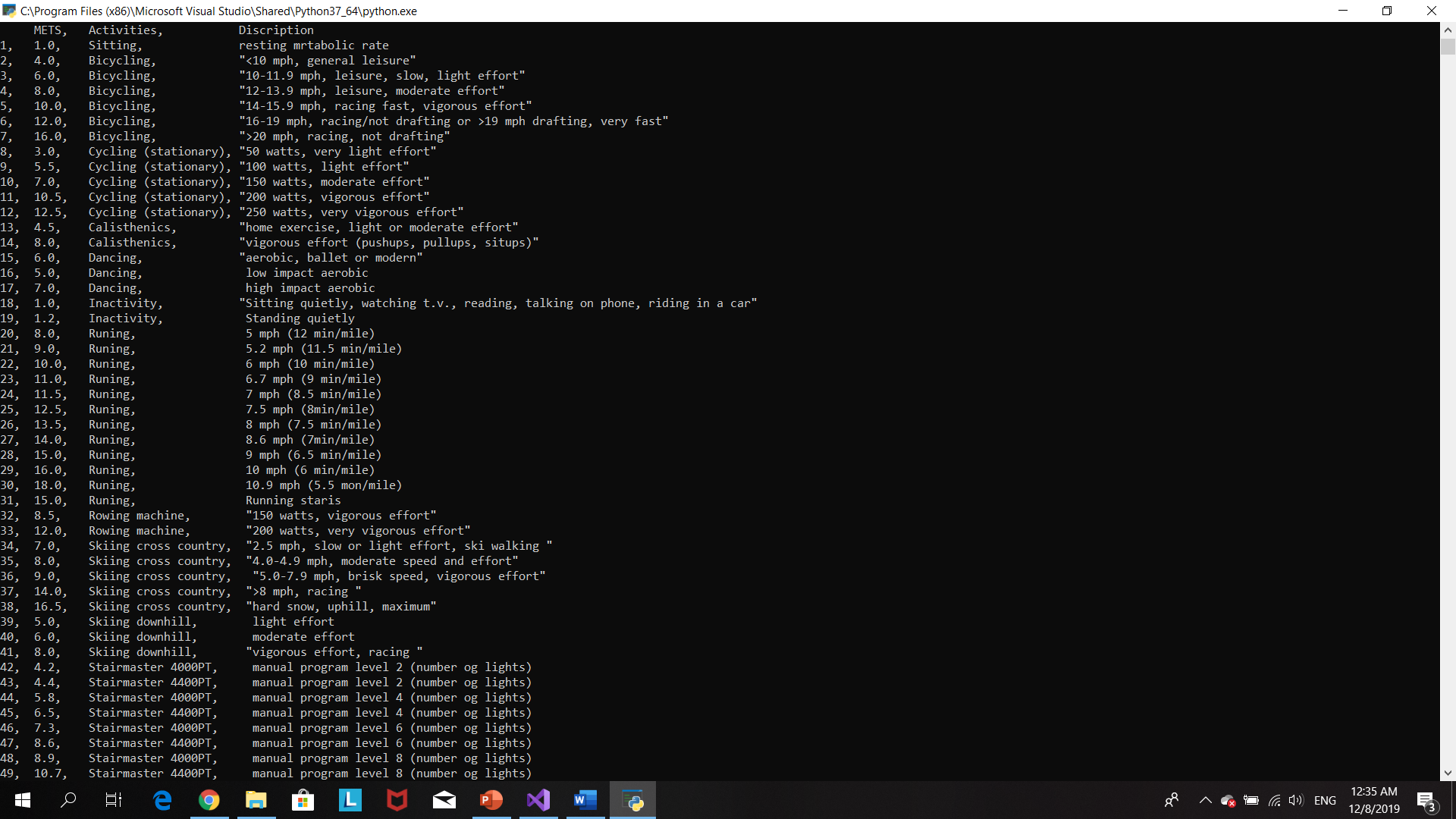
Then program will then print out the user details and ask user to confirm them.



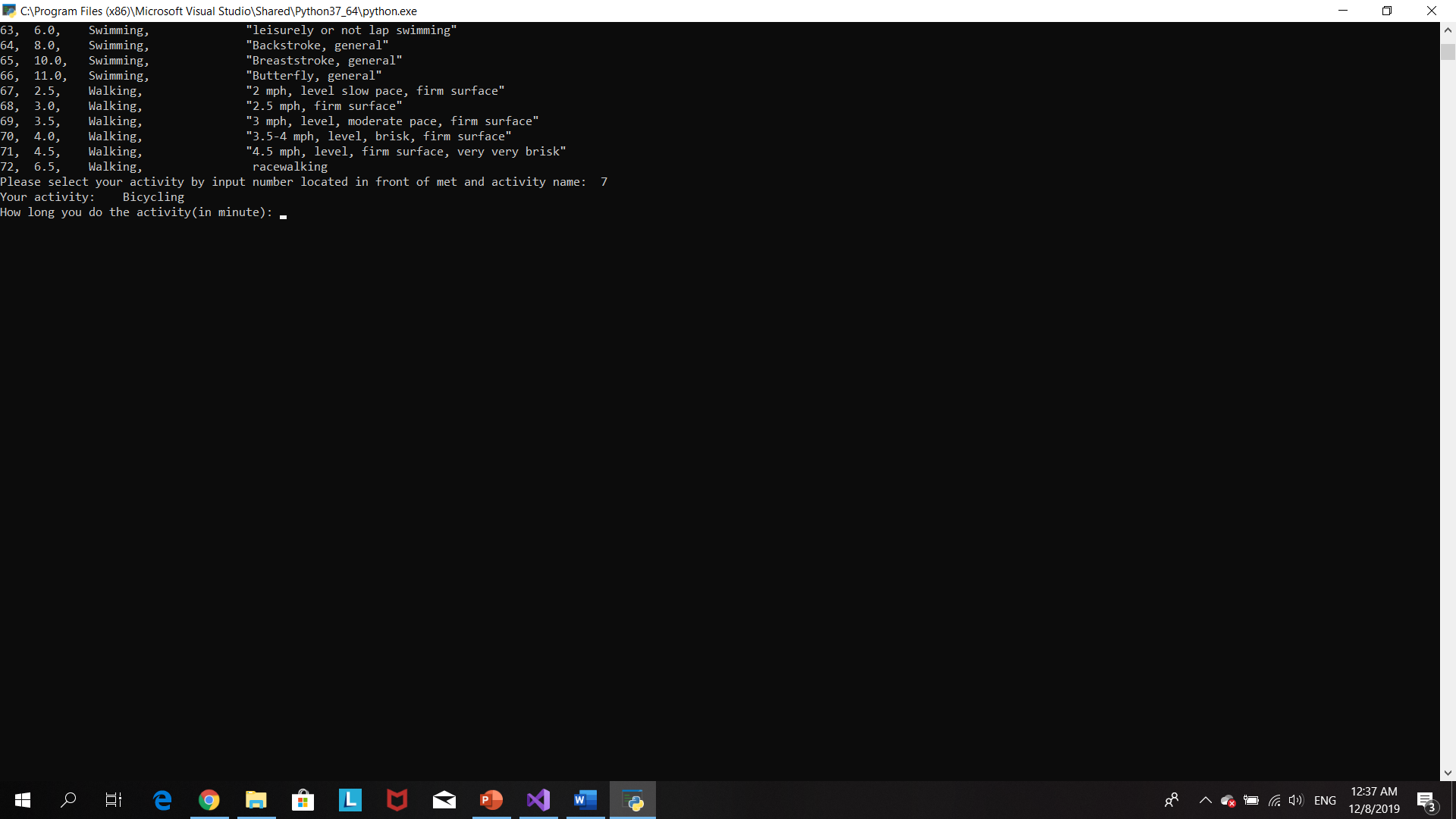
If user what to make a change in their details. He/she has to select ‘n’ for no and program will ask user to input their details again.



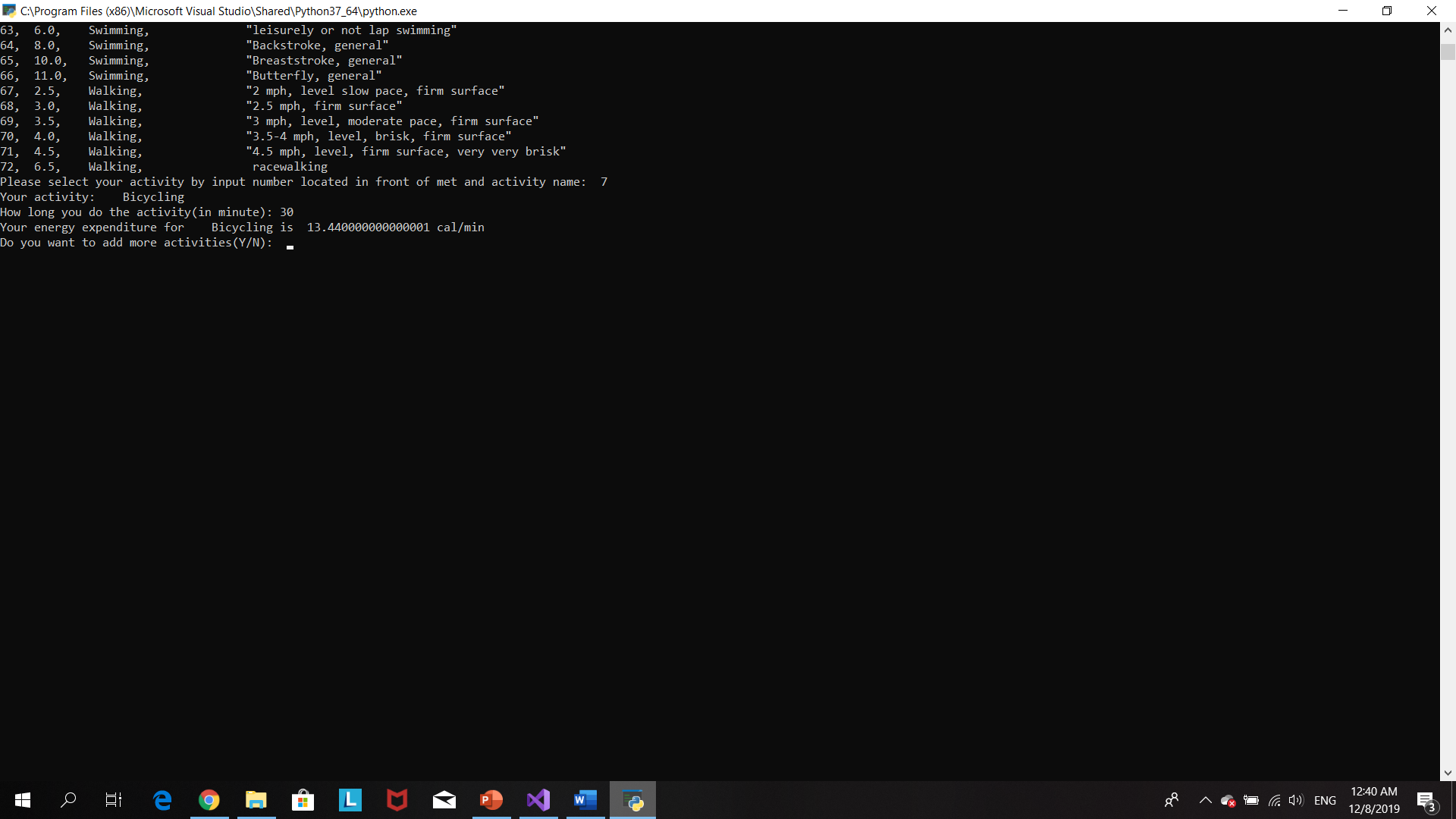
After user confirm his/her personal details, program will print out the activities list



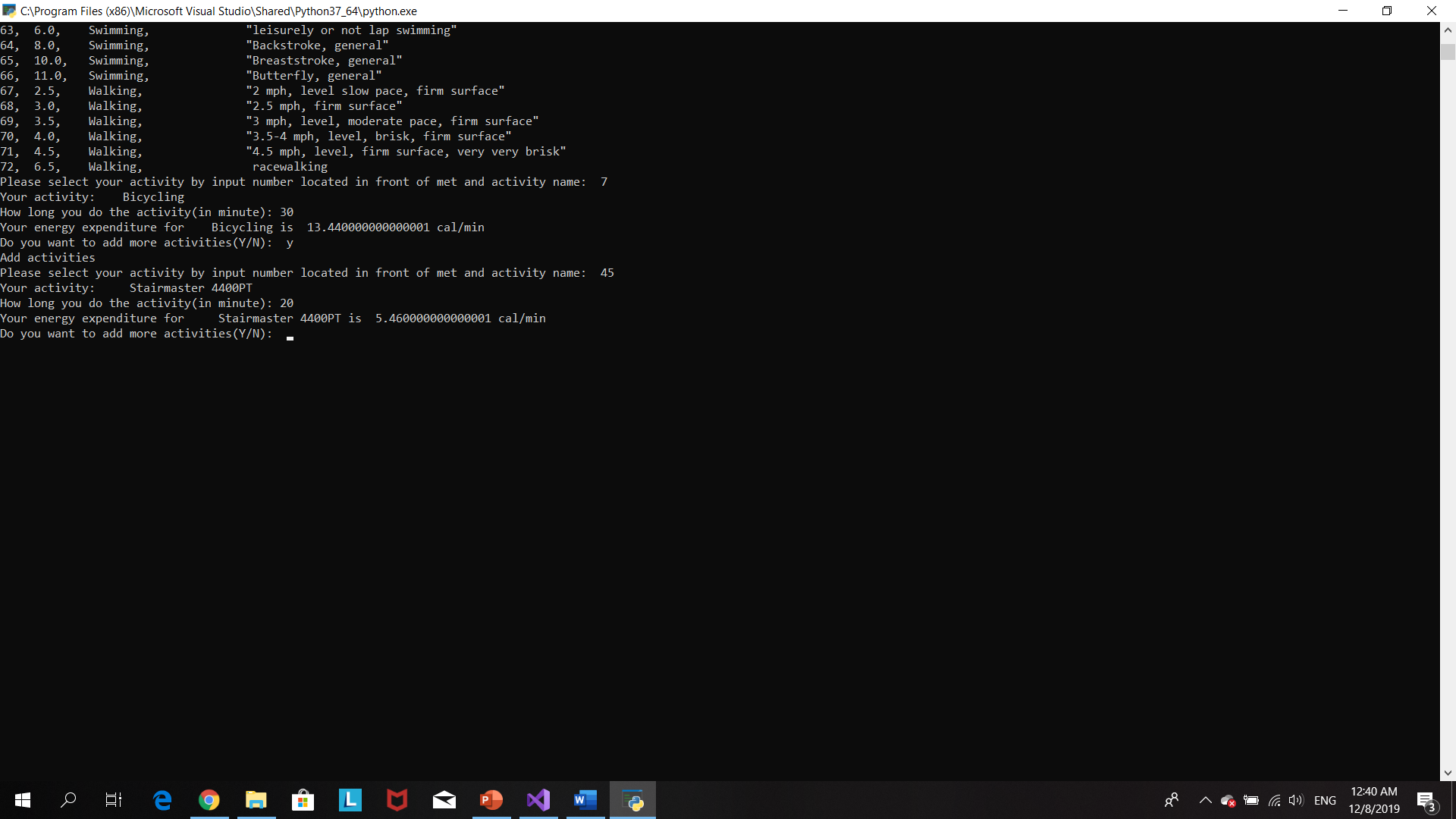
Program ask user to input number indicate activity he/she wants to choose and duration of that activity



Program will calculate the calories burn of the user by doing that activity and ask user whether user want to input more activity or not.

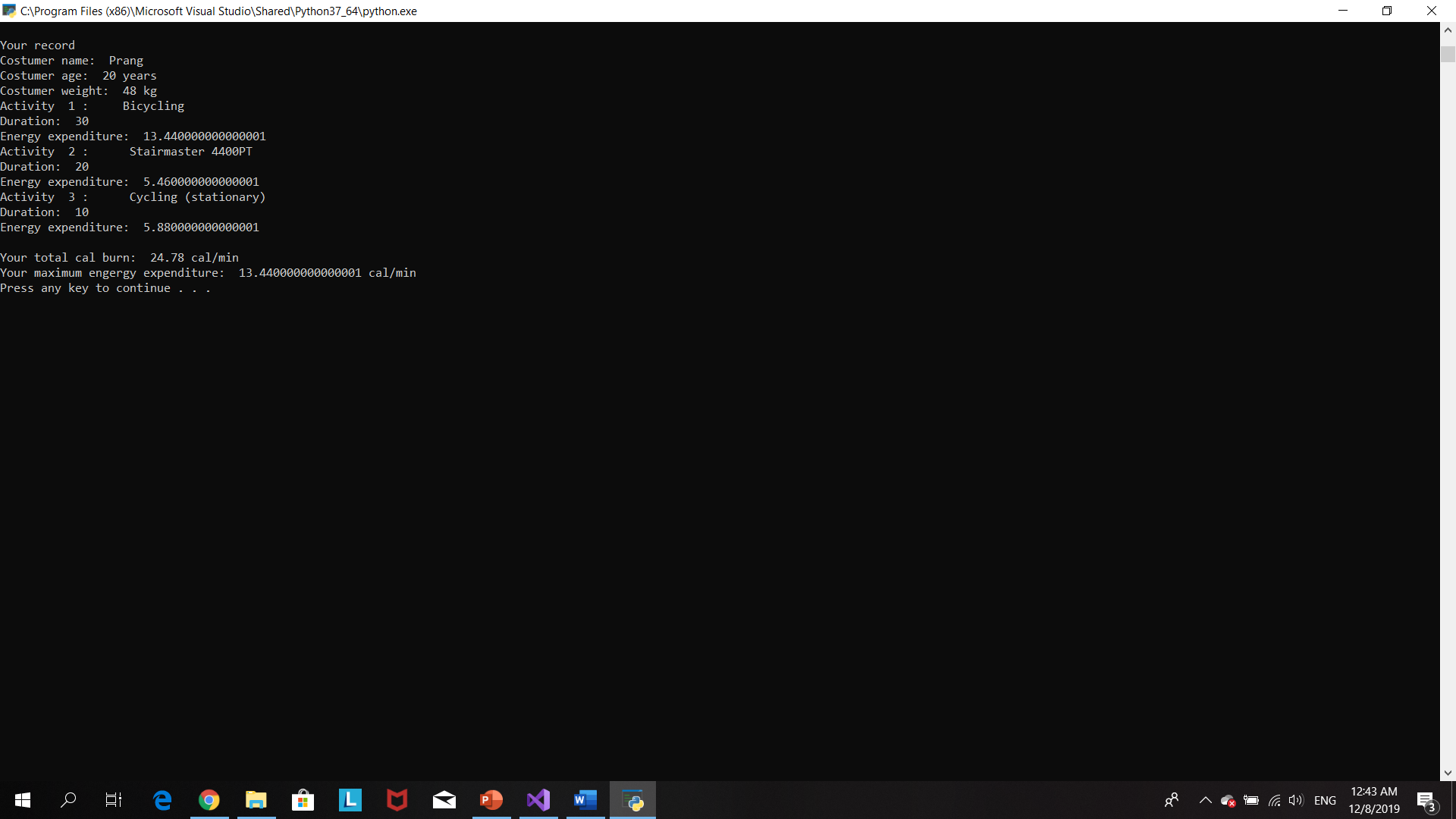


User input more activity



User’s record are printed out. The record including total calories burnt and best performance .

Program stops



# Source code

#The purpose of this profram is to create fitness app used to record the number of calories burned during different kinds of exercise.

#Author: Prang Kongthongluck

#Version: 2.0

#Date: 06/12/2019

age=()

weight=()

mets=()

activities=[]

energy=()

energylist=[]

durationlist=[]

def user():

global weight

global name

global age

#The porpuse of this loop is to allow user to confirm their datails

while True:

name=input("Please enter your name: ")

#make user naem start with alphabet

name=name.capitalize()

age=input("Please enter your age: ")

#make sure user input interger

while True:

try:

weight=int(input("Please enter your weight: "))

break

except ValueError:

print("Please input weight as integer")

#purpose of this loop is to make sure user input collect input to weightUnit

while True:

weightUnit=input("Please choose unite of your weight you entered (P=pound/KG): ")

weightUnit= weightUnit.upper()

#convert weight from pounds in kg when user input weight in pounds)

if (weightUnit=="P"):

weight=weight\*0.45

print("your weight in kg: ",weight)

break

elif (weightUnit=="KG"):

break

else:

print("Invalid input.PLease try again")

print ("Costumer name: ",name)

print("Costumer age: ",age,"years")

print ("Costumer weight: ",weight,"kg")

confirmDetails=input("Confirm your information (Y/N)")

confirmDetails=confirmDetails.upper()

if confirmDetails=="Y":

print("your informations are confirmed")

break

elif confirmDetails=="N":

print("Please re-enter your details")

user()

def MET():

global selected

global act

global energy

global duration

try:

file= open ("mettable.txt")

except:

print("Something went wrong when writing to the file")

#use file to print out list of acttivites

print(file.read())

file.close()

anothergo="Y"

#the purpose of this loop is to allow user to add more avtivity

while (anothergo=="Y"):

check="Y"

#check whether costumer enter right number for activity

while check=="Y":

#make sure user input interger

while True:

try:

userselect=int(input("Please select your activity by input number located in front of met and activity name: "))

break

except ValueError:

print ("Please input with integer")

#split each column in file

with open("mettable.txt") as file:

for i in file:

column=i.split(",")

selected=column[0]

met=column[1]

act=column[2]

if str(userselect)==selected:

print("Your activity:",act)

activities.append(act)

#make sure user input interger

while True:

try:

duration=int(input("How long you do the activity(in minute): "))

break

except ValueError:

print ("Please input with integer")

durationlist.append(duration)

#Energy expenditure calculation

energy=float(0.0175\*(float(met))\* weight)

#add energy in to energy list to help calculate total energ at the end

energylist.append(energy)

print("Your energy expenditure for",act,"is ",energy,"cal/min")

if (userselect <0) or (userselect>72):

print("Incorrect input, please try again")

elif (userselect >0) or (userselect<=72):

break

else:

print("Incorrect input, please try again")

anothergo=input("Do you want to add more activities(Y/N): ")

anothergo=anothergo.upper()

if (anothergo=="Y"):

print("Add activities")

elif (anothergo=="N"):

break

else:

print("Invalid input. Please try again")

MET()

#this function will help print out activity summary

def activity(a,b,c):

x=0

num=1

for i in range(len(activities)):

print("Activity ",num,": ",a[x])

print("Duration: ",b[x])

print("Energy expenditure: ",c[x])

x +=1

num +=1

def summary():

print("")

print("")

print("Your record")

print ("Costumer name: ",name)

print("Costumer age: ",age,"years")

print ("Costumer weight: ",weight,"kg")

#call function activity

activity(activities,durationlist,energylist)

totalcal=sum(energylist)

print("")

print("Your total cal burn: ",totalcal,"cal/min")

#max() print out the maximum value in the list

print("Your maximum engergy expenditure: ",max(energylist),"cal/min")

summary()

# Reference

101Computing (2017). *Python Tutorial - Takeaway Ordering System*. Available at: https://www.youtube.com/watch?v=IW\_IASFhmtA

Fincher, J. (2018). *Reading and Writing CSV Files in Python – Real Python*. [online] Realpython.com. Available at: https://realpython.com/python-csv/

GeeksforGeeks. (2017). *Working with csv files in Python - GeeksforGeeks*. [online] Available at: https://www.geeksforgeeks.org/working-csv-files-python/

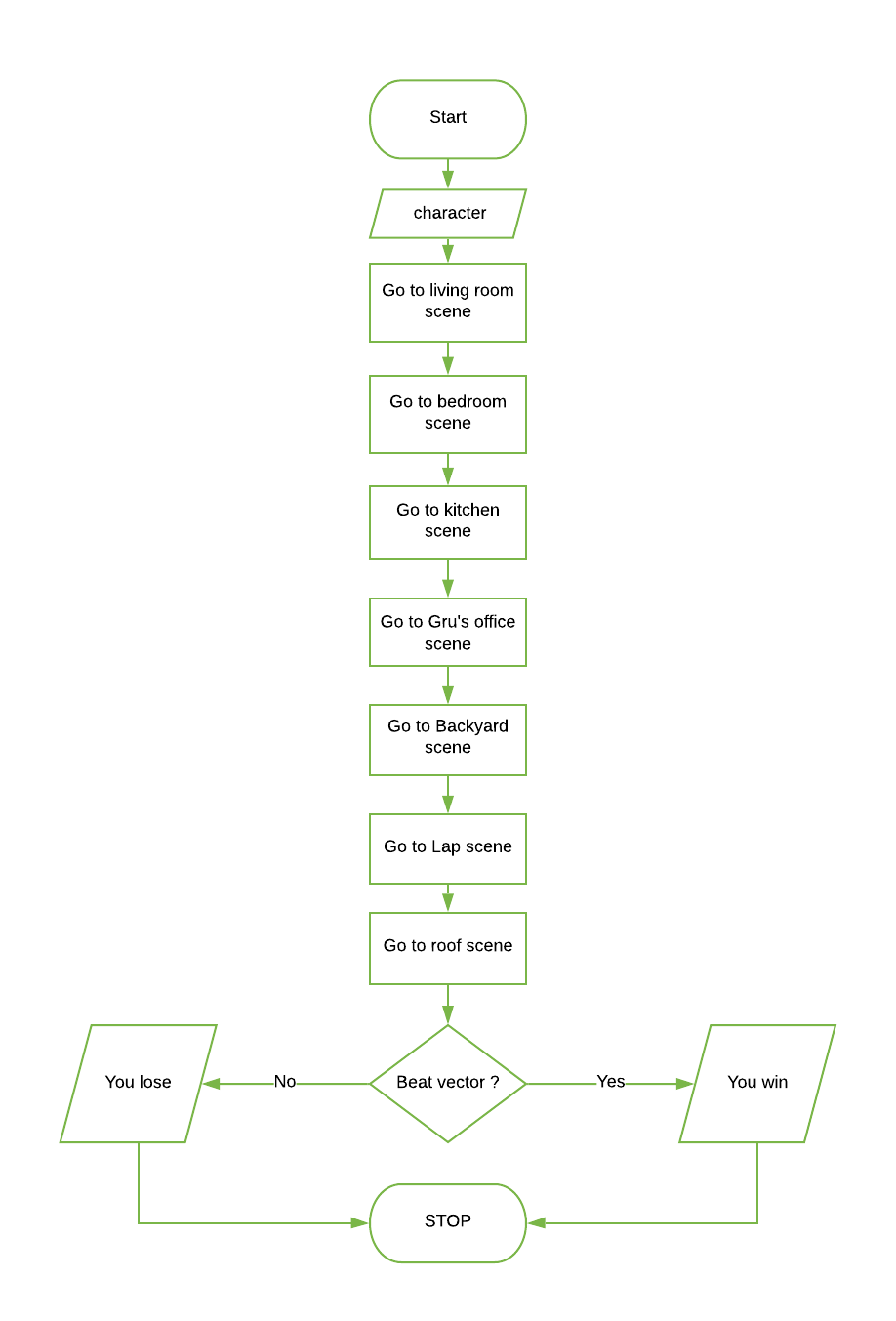
Nethala, A. (2017). *Python program to find the largest and smallest number in a list*. [online] Medium. Available at: https://medium.com/programminginpython-com/python-program-to-find-the-largest-and-smallest-number-in-a-list-fd8fac8aba08

W3schools.com. (2019). *Python For Loops*. [online] Available at: https://www.w3schools.com/python/python\_for\_loops.asp

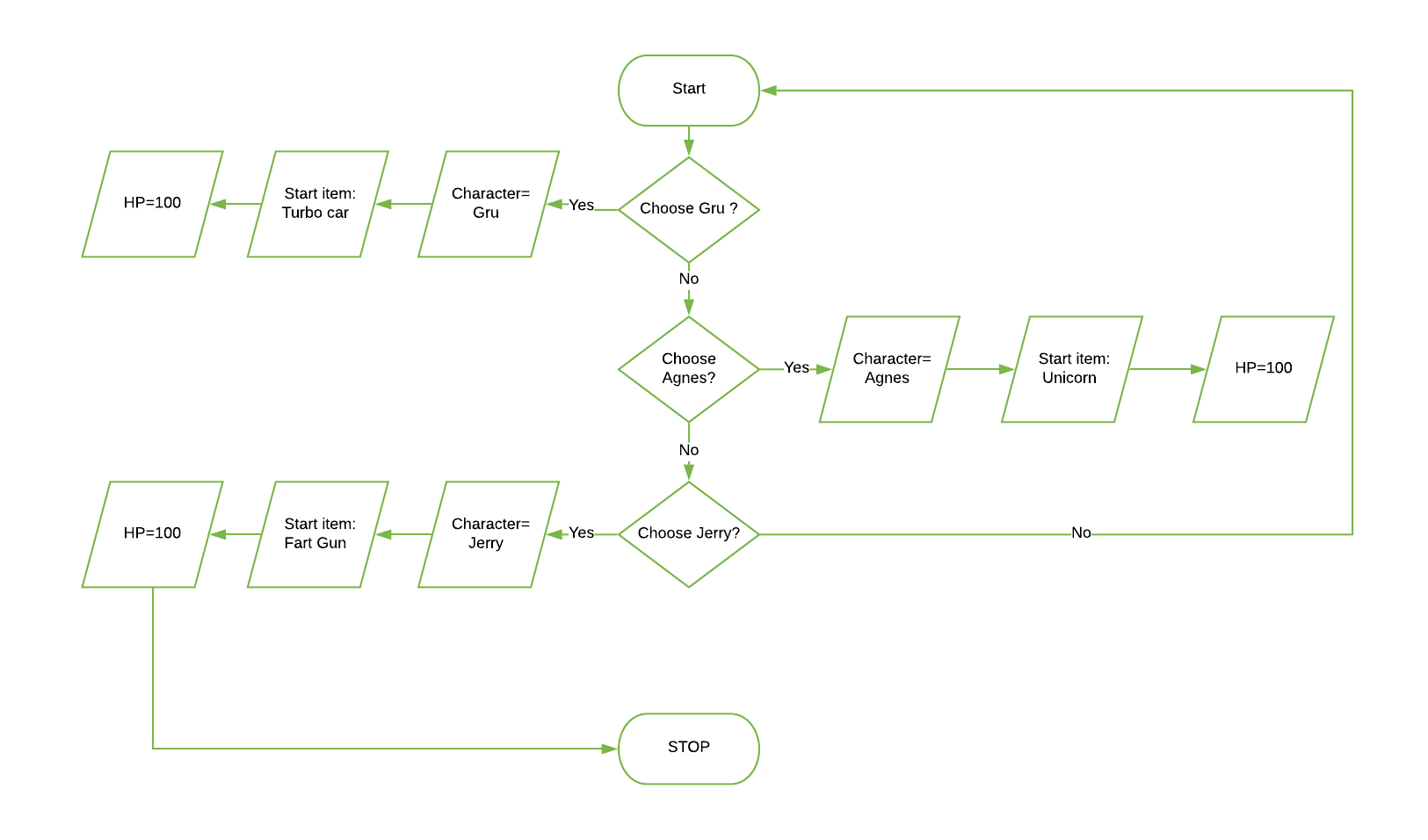
Program 3 : Fantasy Adventure Game Report

# 

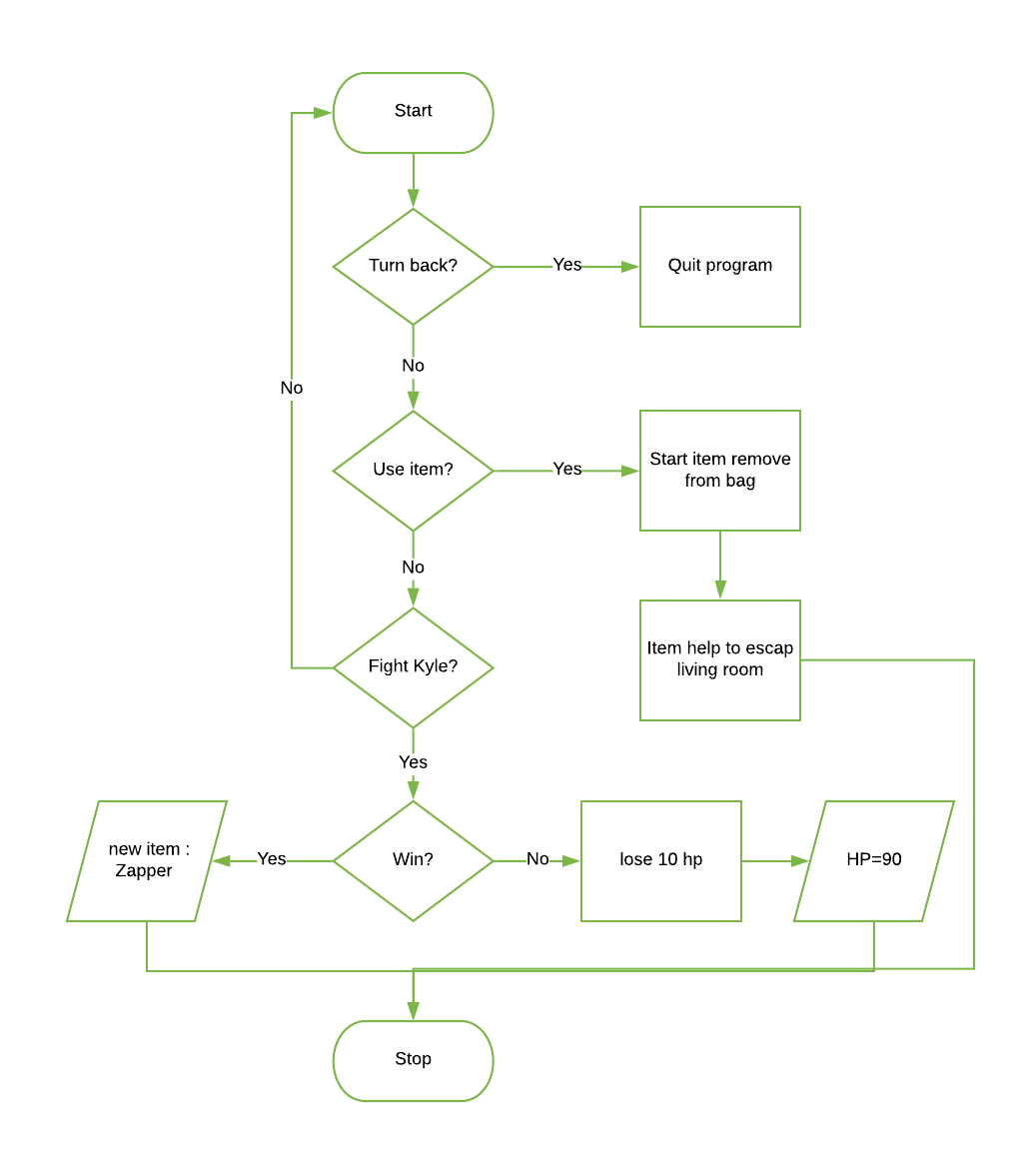
# Fantasy adventure game program flow chart



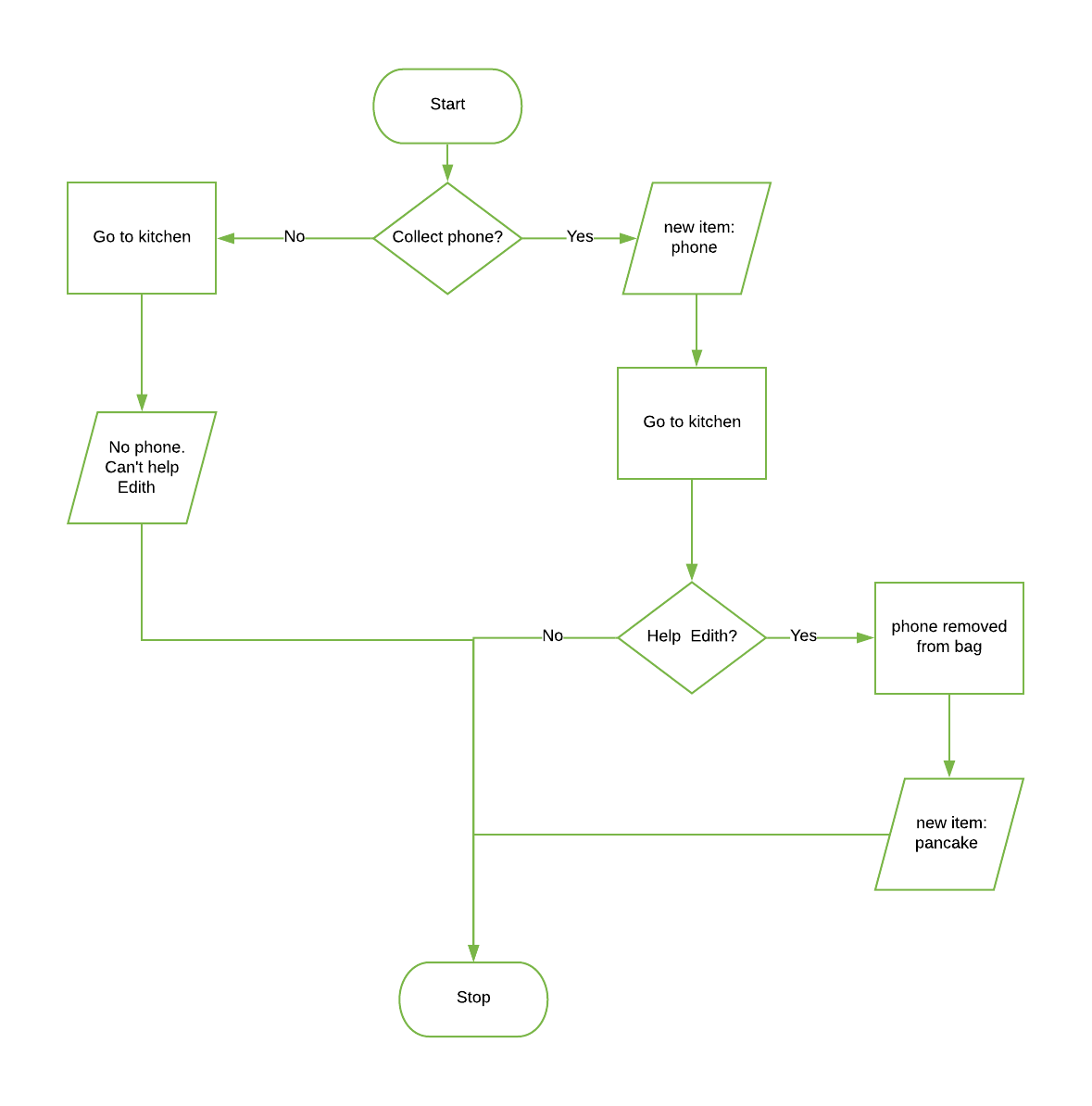
## Subprocess: Choose character



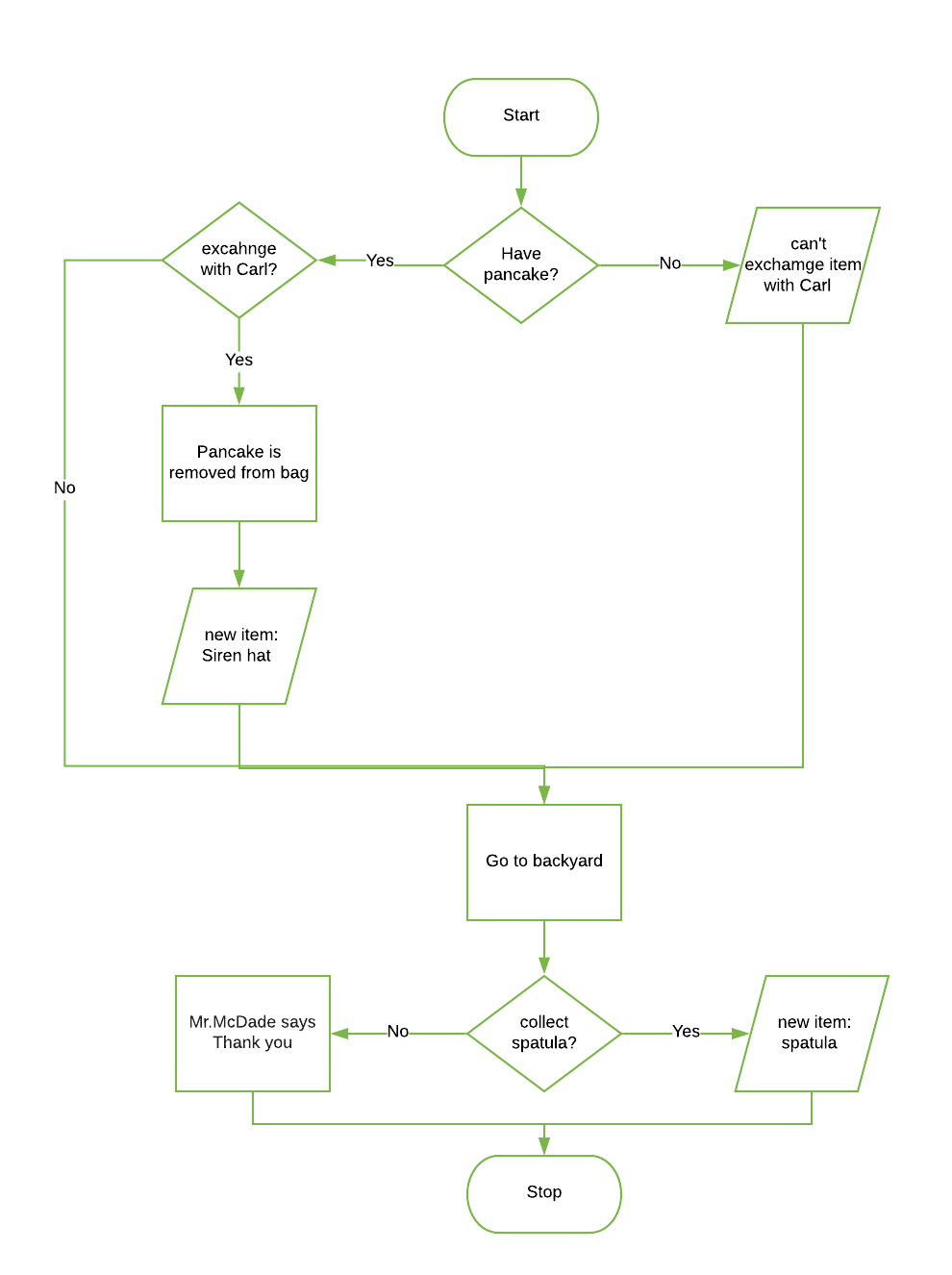
## Sunprocess: Living room scene



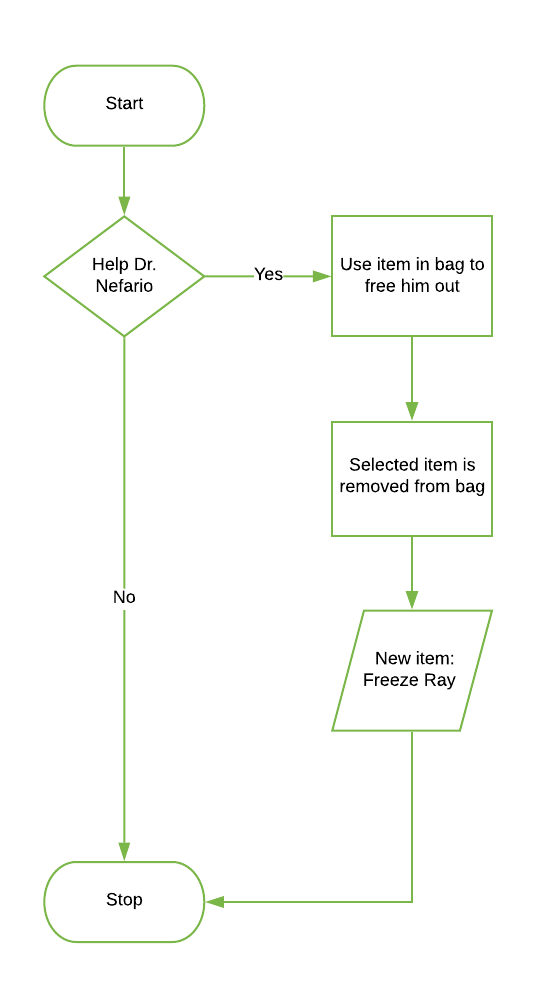
## Subprocess: Bedroom and Kitchen



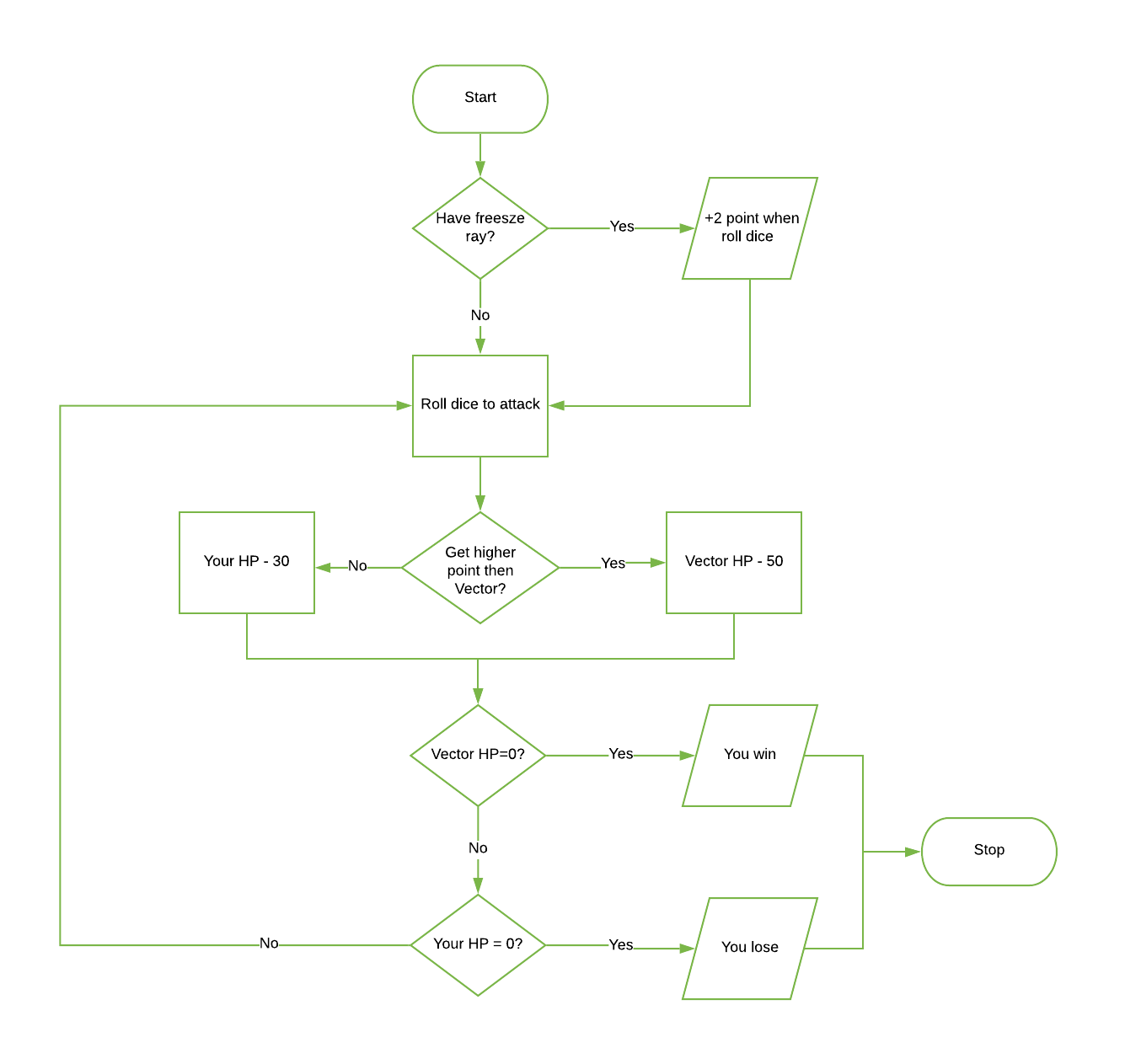
## Subprocess: Gru’s office and backyard



## Subprocess: Lap scene



## Sunprocesse: Roof scene

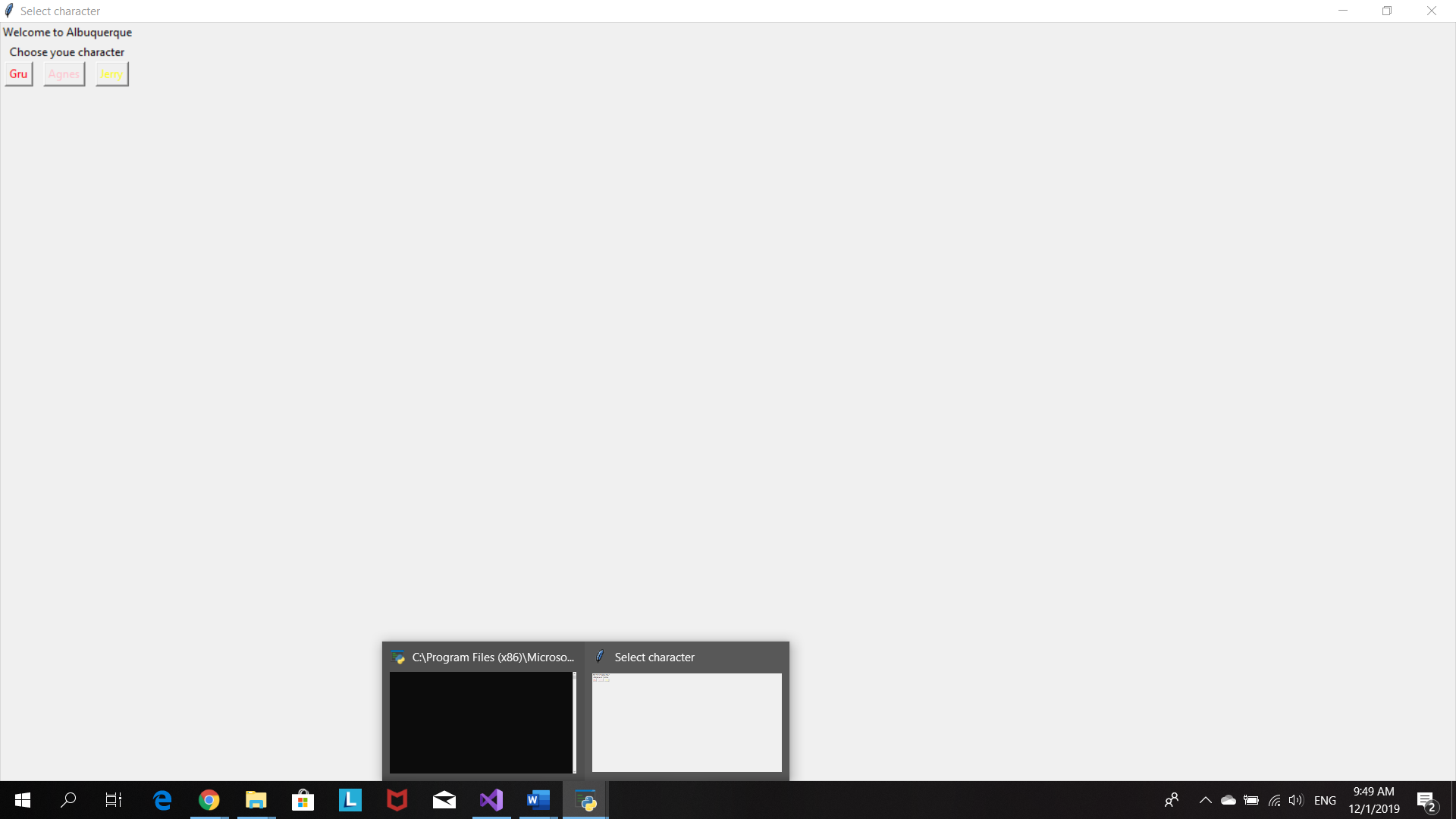


# Testing Table- Program 3: Fantasy adventure game

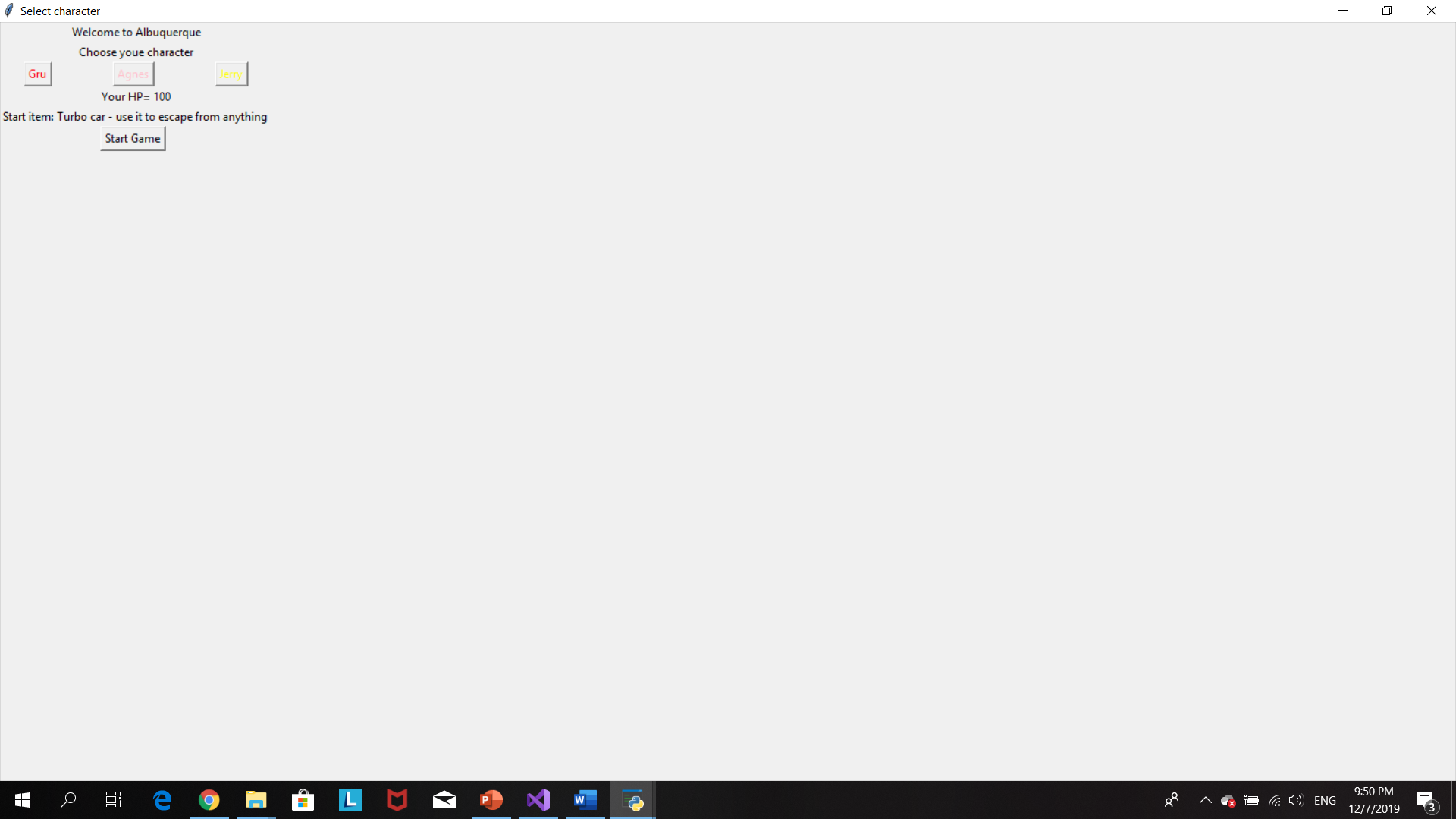
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. | Item to test | Test description | Test action | Expected result | Actual result | Comment |
| 1 | Button(root,text="Gru",fg="red",command=Gruitem) | Valid data | Click Button | Text and button pop up | Text display and start button pop up | - |
| 2 | Button(root,text="Gru",fg="red",command=Gruitem) | Valid command function | command=Gruitem | Text and button pop up | Text display and start button pop up | - |
| 3 | Button(root,text="Gru",fg="red",command=Gruitem) | Function is renamed | def whatwhat(): | Text and button pop up | Program crashed-command function do not define | Build back-up command function |
| 4 | Button(root,text="Gru",fg="red",command=Gruitem) | Function doesn’t not exit | Delete function Gruitem | Text and button pop up | Program crashed- function do not define | Build back-up command function |
| 5 | Button(root,text="Agnes",fg="pink",command=Agnesitem) | Valid data | Click Button | Text and button pop up | Text display and start button pop up | - |
| 6 | Button(root,text="Agnes",fg="pink",command=Agnesitem) | Valid command function | command= Agnesitem | Text and button pop up | Text display and start button pop up | - |
| 7 | Button(root,text="Agnes",fg="pink",command=Agnesitem) | Function is renamed | def whatwhat(): | Text and button pop up | Program crashed-command function do not define | Build back-up command function |
| 8 | Button(root,text="Agnes",fg="pink",command=Agnesitem) | Function doesn’t not exit | Delete function Agnesitem | Text and button pop up | Program crashed- function do not define | Build back-up command function |
| 9 | Button(root,text="Jerry",fg="yellow",command=Jerryitem) | Valid data | Click Button | Text and button pop up | Text display and start button pop up | - |
| 10 | Button(root,text="Jerry",fg="yellow",command=Jerryitem) | Valid command function | command= Jerryitem | Text and button pop up | Text display and start button pop up | - |
| 11 | Button(root,text="Jerry",fg="yellow",command=Jerryitem) | Function is renamed | def whatwhat(): | Text and button pop up | Program crashed-command function do not define | Build back-up command function |
| 12 | Button(root,text="Jerry",fg="yellow",command=Jerryitem) | Function doesn’t not exit | Delete function Jerryitem | Text and button pop up | Program crashed- function do not define | Build back-up command function |
| 13 | Button(root,text="Start Game",command=HouseGru) | Valid data | Click Button | New window pop up | New window display – program follow the command | - |
| 14 | Button(root,text="Start Game",command=HouseGru) | Valid command function | command=HouseGru | New window pop up | New window display – program follow the command | - |
| 15 | Button(root,text="Start Game",command=HouseGru) | Function is renamed | def whatwhat(): | New window pop up | Program crashed-command function do not define | Build back-up command function |
| 16 | Button(root,text="Start Game",command=HouseGru) | Function doesn’t not exit | Delete function HouseGru | New window pop up | Program crashed- function do not define | Build back-up command function |
| 17 | Button(house,text="Turn back",command=quit) | Valid data | Click Button | Quit program | Program crash | Fix the command ‘quit’ |
| 18 | Button(house,text="Turn back",command=quit) | Valid command | command= quit | Quit program | Program crash | Fix the command ‘quit’ |
| 19 | Button(house,text="Turn back",command=quit) | Invalid command | Command=QuIt | Quit program | Program crash | Fix the command ‘quit’ |
| 20 | Button(house,text="fight",command=(fight)) | Valid data | Click Button | Labels and button pop up | Labels and button pop up – program continue as normal | - |
| 21 | Button(house,text="fight",command=(fight)) | Valid command function | Command= fight | Labels and button pop up | Labels and button pop up – program continue as normal | - |
| 22 | Button(house,text="fight",command=(fight)) | Function is renamed | def whatwhat(): | Labels and button pop up | Program crashed- function do not define | Double check spelling |
| 23 | Button(house,text="fight",command=(fight)) | Function doesn’t not exit | Delete function fight | Labels and button pop up | Program crashed- function do not define | Double check command |
| 24 | Button(house,text="Use item",command=houseitemGru) | Valid data | Click Button | Program follow command | start item is removed from the item bag and labels pop up | - |
| 25 | Button(house,text="Use item",command=houseitemGru) | Valid command function | command= houseitemGru | Program follow command | start item is removed from the item bag and labels pop up | - |
| 26 | Button(house,text="Use item",command=houseitemGru) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 27 | Button(house,text="Use item",command=houseitemGru) | Function doesn’t not exit | Delete function  houseitemGru | Error massage | Program crashed- function do not define | Double check command |
| 28 | Button(house,text="continue",command=bedroom) | Valid data | Click Button | Program follow command | New window pop up – program continue as normal | - |
| 29 | Button(house,text="continue",command=bedroom) | Valid command function | command=bedroom | Program follow command | New window pop up – program continue as normal | - |
| 30 | Button(house,text="continue",command=bedroom) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 31 | Button(house,text="continue",command=bedroom) | Function doesn’t not exit | Delete function  bedroom | Error massage | Program crashed- function do not define | Double check command |
| 32 | Button(bed,text="Leave the room",anchor="e",command=Kitchen) | Valid data | Click Button | Program follow command | New window pop up – program follow command | - |
| 33 | Button(bed,text="Leave the room",anchor="e",command=Kitchen) | Valid command function | command= Kitchen | Program follow command | New window pop up – program follow command | - |
| 34 | Button(bed,text="Leave the room",anchor="e",command=Kitchen) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 35 | Button(bed,text="Leave the room",anchor="e",command=Kitchen) | Function doesn’t not exit | Delete function  Kitchen | Error massage | Program crashed- function do not define | Double check command |
| 36 | Button(bed,text="collect Margo phone",anchor="w",command=MargoPhone) | Valid data | Click Button | Program follow command | Phone add to item list, labels display - program run as normal | - |
| 37 | Button(bed,text="collect Margo phone",anchor="w",command=MargoPhone) | Valid command function | command=MargoPhone | Program follow command | Phone add to item list, labels display - program run as normal | - |
| 38 | Button(bed,text="collect Margo phone",anchor="w",command=MargoPhone) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 39 | Button(bed,text="collect Margo phone",anchor="w",command=MargoPhone) | Function doesn’t not exit | Delete function  MargoPhone | Error massage | Program crashed- function do not define | Double check command |
| 40 | Button(bed,text="continue",anchor="e",command=Kitchen) | Valid data | Click Button | Program follow command | New window pop up – program follow command | - |
| 41 | Button(bed,text="continue",anchor="e",command=Kitchen) | Valid command function | command= Kitchen | Program follow command | New window pop up – program follow command | - |
| 42 | Button(bed,text="continue",anchor="e",command=Kitchen) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 43 | Button(bed,text="continue",anchor="e",command=Kitchen) | Function doesn’t not exit | Delete function  Kitchen | Error massage | Program crashed- function do not define | Double check command |
| 44 | Button(kit,text="help Edith",command=HelpEdith) | Valid data | Click Button | Program follow command | List box pop up – program run as normal | - |
| 45 | Button(kit,text="help Edith",command=HelpEdith) | Valid command function | command=HelpEdith | Program follow command | List box pop up – program run as normal | - |
| 46 | Button(kit,text="help Edith",command=HelpEdith) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 47 | Button(kit,text="help Edith",command=HelpEdith) | Function doesn’t not exit | Delete function  HelpEdith | Error massage | Program crashed- function do not define | Double check command |
| 48 | Button(kit,text="dont't have phone",command=GruRoom) | Valid data | Click Button | Program follow command | New window (Gru’s office) pop up- program follow command | - |
| 49 | Button(kit,text="dont't have phone",command=GruRoom) | Valid command function | command=GruRoom | Program follow command | New window (Gru’s office) pop up- program follow command | - |
| 50 | Button(kit,text="dont't have phone",command=GruRoom) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 51 | Button(kit,text="dont't have phone",command=GruRoom) | Function doesn’t not exit | Delete function GruRoom | Error massage | Program crashed- function do not define | Double check command |
| 52 | Button(kit,text="Continue",command=GruRoom) | Valid data | Click Button | Program follow command | New window (Gru’s office) pop up- program follow command | - |
| 53 | Button(kit,text="Continue",command=GruRoom) | Valid command function | command=GruRoom | Program follow command | New window (Gru’s office) pop up- program follow command | - |
| 54 | Button(kit,text="Continue",command=GruRoom) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 55 | Button(kit,text="Continue",command=GruRoom) | Function doesn’t not exit | Delete function GruRoom | Error massage | Program crashed- function do not define | Double check command |
| 56 | Button(kit,text="Select",command=selectedItemHelpEdith) | Valid data | Click Button | Program follow command | Phone is removed from list and Pancake is added to list | Selected item is removed from the list |
| 57 | Button(kit,text="Select",command=selectedItemHelpEdith) | Valid command function | command= selectedItemHelpEdith | Program follow command | Phone is removed from list and Pancake is added to list | Selected item is removed from the list |
| 58 | Button(kit,text="Select",command=selectedItemHelpEdith) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Error massage |
| 59 | Button(kit,text="Select",command=selectedItemHelpEdith) | Function doesn’t not exit | Delete function selectedItemHelpEdith | Error massage | Program crashed- function do not define | Error massage |
| 60 | Button(gru,text="Exchange",command=ExchangeCarl) | Valid data | Click Button | Program follow command | Pancake is removed from list and Siren hat is added to list | Selected item is removed from the list |
| 61 | Button(gru,text="Exchange",command=ExchangeCarl) | Valid command function | command= ExchangeCarl | Program follow command | Pancake is removed from list and Siren hat is added to list | Selected item is removed from the list |
| 62 | Button(gru,text="Exchange",command=ExchangeCarl) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage |
| 63 | Button(gru,text="Exchange",command=ExchangeCarl) | Function doesn’t not exit | Delete function  ExchangeCarl | Error massage | Program crashed- function do not define | Display Error massage |
| 64 | Button(gru,text="Don't have pancake", command=Backyard) | Valid data | Click Button | Program follow command | New window (Backyard)pop up – program follow command | - |
| 65 | Button(gru,text="Don't have pancake", command=Backyard) | Valid command function | command= Backyard | Program follow command | New window (Backyard)pop up – program follow command | - |
| 66 | Button(gru,text="Don't have pancake", command=Backyard) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage  Double check spelling |
| 67 | Button(gru,text="Don't have pancake", command=Backyard) | Function doesn’t not exit | Delete function  Backyard | Error massage | Program crashed- function do not define | Display Error massage |
| 68 | Button(gru,text="Continue",command=Backyard) | Valid data | Click Button | Program follow command | New window (Backyard)pop up – program follows command | - |
| 69 | Button(gru,text="Continue",command=Backyard) | Valid command function | command= Backyard | Program follow command | New window (Backyard)pop up – program follows command | - |
| 70 | Button(gru,text="Continue",command=Backyard) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage  Double check spelling |
| 71 | Button(gru,text="Continue",command=Backyard) | Function doesn’t not exit | Delete function  Backyard | Error massage | Program crashed- function do not define | Display Error massage |
| 72 | Button(back,text="return the spatular",command=returnSpatula) | Valid data | Click Button | Program follow command | Label pop up – program continue as normal | - |
| 73 | Button(back,text="return the spatular",command=returnSpatula) | Valid command function | command= returnSpatula | Program follow command | Label pop up – program continue as normal | - |
| 74 | Button(back,text="return the spatular",command=returnSpatula) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage |
| 75 | Button(back,text="return the spatular",command=returnSpatula) | Function doesn’t not exit | Delete function  returnSpatula | Error massage | Program crashed- function do not define | Display Error massage |
| 76 | Button(back,text="Keep the spactular",command=keepSpatulat) | Valid data | Click Button | Program follow command | Spatula is added to list - program follow command | - |
| 77 | Button(back,text="Keep the spactular",command=keepSpatulat) | Valid command function | command= returnSpatula | Program follow command | Spatula is added to list - program follow command | - |
| 78 | Button(back,text="Keep the spactular",command=keepSpatulat) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage |
| 79 | Button(back,text="Keep the spactular",command=keepSpatulat) | Function doesn’t not exit | Delete function  returnSpatula | Error massage | Program crashed- function do not define | Display Error massage |
| 80 | Button(back,text="continue",command=Lab) | Valid data | Click Button | Program follow command | New window (Minion Lab) pop up – program follow command | - |
| 81 | Button(back,text="continue",command=Lab) | Valid command function | command= Lab | Program follow command | New window (Minion Lab) pop up – program follow command | - |
| 82 | Button(back,text="continue",command=Lab) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage  Double check spelling |
| 83 | Button(back,text="continue",command=Lab) | Function doesn’t not exit | Delete function  Lab | Error massage | Program crashed- function do not define | Display Error massage |
| 84 | Button(lab,text="Help him",anchor='w',command=HelpNefario) | Valid data | Click Button | Program follow command | List box pop up – program follow command | - |
| 85 | Button(lab,text="Help him",anchor='w',command=HelpNefario) | Valid command function | command= HelpNefario | Program follow command | List box pop up – program follow command | - |
| 86 | Button(lab,text="Help him",anchor='w',command=HelpNefario) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Double check spelling |
| 87 | Button(lab,text="Help him",anchor='w',command=HelpNefario) | Function doesn’t not exit | Delete function  HelpNefario | Error massage | Program crashed- function do not define | Display Error massage |
| 88 | Button(lab,text="Ignor him",anchor='e',command=Roof) | Valid data | Click Button | Program follow command | New window ( Roof) pop up – program follow command | - |
| 89 | Button(lab,text="Ignor him",anchor='e',command=Roof) | Valid command function | command= Roof | Program follow command | New window ( Roof) pop up – program follow command | - |
| 90 | Button(lab,text="Ignor him",anchor='e',command=Roof) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage |
| 91 | Button(lab,text="Ignor him",anchor='e',command=Roof) | Function doesn’t not exit | Delete function  Roof | Error massage | Program crashed- function do not define | Display Error massage |
| 92 | Button(lab, text = "select",command=print\_HelpNafario) | Valid data | Click Button | Program follow command | Label pop out when each item is selected | - |
| 93 | Button(lab, text = "select",command=print\_HelpNafario) | Valid command function | command= print\_HelpNafario | Program follow command | Label pop out when each item is selected | - |
| 94 | Button(lab, text = "select",command=print\_HelpNafario) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage   Build back-up command function |
| 95 | Button(lab, text = "select",command=print\_HelpNafario) | Function doesn’t not exit | Delete function  print\_HelpNafario) | Error massage | Program crashed- function do not define | Display Error massage   Build back-up command function |
| 96 | Button(lab,text="Continue",command=RoofHelp) | Valid data | Click Button | Program follow command | New window pop up – program follow command | - |
| 97 | Button(lab,text="Continue",command=RoofHelp) | Valid command function | command= RoofHelp | Program follow command | New window pop up – program follow command | - |
| 98 | Button(lab,text="Continue",command=RoofHelp) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage |
| 99 | Button(lab,text="Continue",command=RoofHelp) | Function doesn’t not exit | Delete function  RoofHelp | Error massage | Program crashed- function do not define | Display Error massage |
| 100 | Button(roof,text="roll dice",command=rollDiceVector) | Valid data | Click Button | Program follow command | Program continue as normal | - |
| 101 | Button(roof,text="roll dice",command=rollDiceVector) | Valid command function | command= rollDiceVector | Program follow command | Program continue as normal | - |
| 102 | Button(roof,text="roll dice",command=rollDiceVector) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage |
| 103 | Button(roof,text="roll dice",command=rollDiceVector) | Function doesn’t not exit | Delete function  rollDiceVector | Error massage | Program crashed- function do not define | Display Error massage |
| 104 | Button(roofH,text="Use Freez Ray",command=FreezRay) | Valid data | Click Button | Program follow command | Labels and Button pop up – program follow command | - |
| 105 | Button(roofH,text="Use Freez Ray",command=FreezRay) | Valid command function | command= FreezRay | Program follow command | Labels and Button pop up – program follow command | - |
| 106 | Button(roofH,text="Use Freez Ray",command=FreezRay) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage |
| 107 | Button(roofH,text="Use Freez Ray",command=FreezRay) | Function doesn’t not exit | Delete function  FreezRay | Error massage | Program crashed- function do not define | Display Error massage |
| 108 | Button(roofH,text="roll dice",command=rollDiceVectorH) | Valid data | Click Button | Program follow command | +2 point on number get from rolling dice | - |
| 109 | Button(roofH,text="roll dice",command=rollDiceVectorH) | Valid command function | command= rollDiceVectorH | Program follow command | +2 point on number get from rolling dice | - |
| 110 | Button(roofH,text="roll dice",command=rollDiceVectorH) | Function is renamed | def whatwhat(): | Error massage | Program crashed- function do not define | Display Error massage |
| 111 | Button(roofH,text="roll dice",command=rollDiceVectorH) | Function doesn’t not exit | Delete function  rollDiceVectorH | Error massage | Program crashed- function do not define | Display Error massage |
| 112 | Button(roof,text="Exite Game",command=quit) | Valid data | Click Button | Quit program | Program crash | Fix the command ‘quit’ |
| 113 | Button(roof,text="Exite Game",command=quit) | Valid command | command= quit | Quit program | Program crash | Fix the command ‘quit’ |
| 114 | Button(roof,text="Exite Game",command=quit) | Invalid command | Command=QuIt | Quit program | Program crash | Fix the command ‘quit’ |

# Screen shot

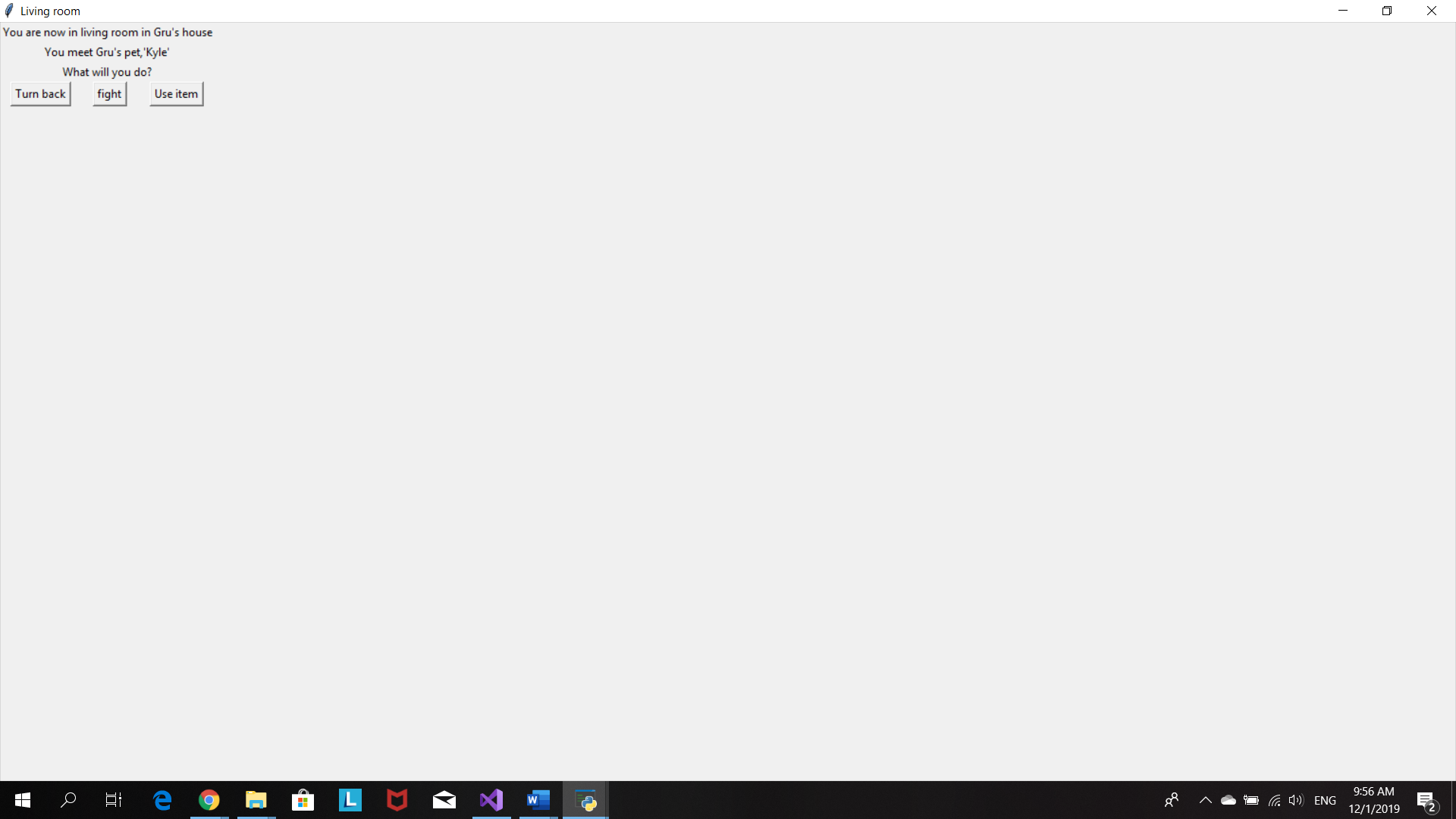
Select character window show up when run the program



Click on button Gru to choose him as player character

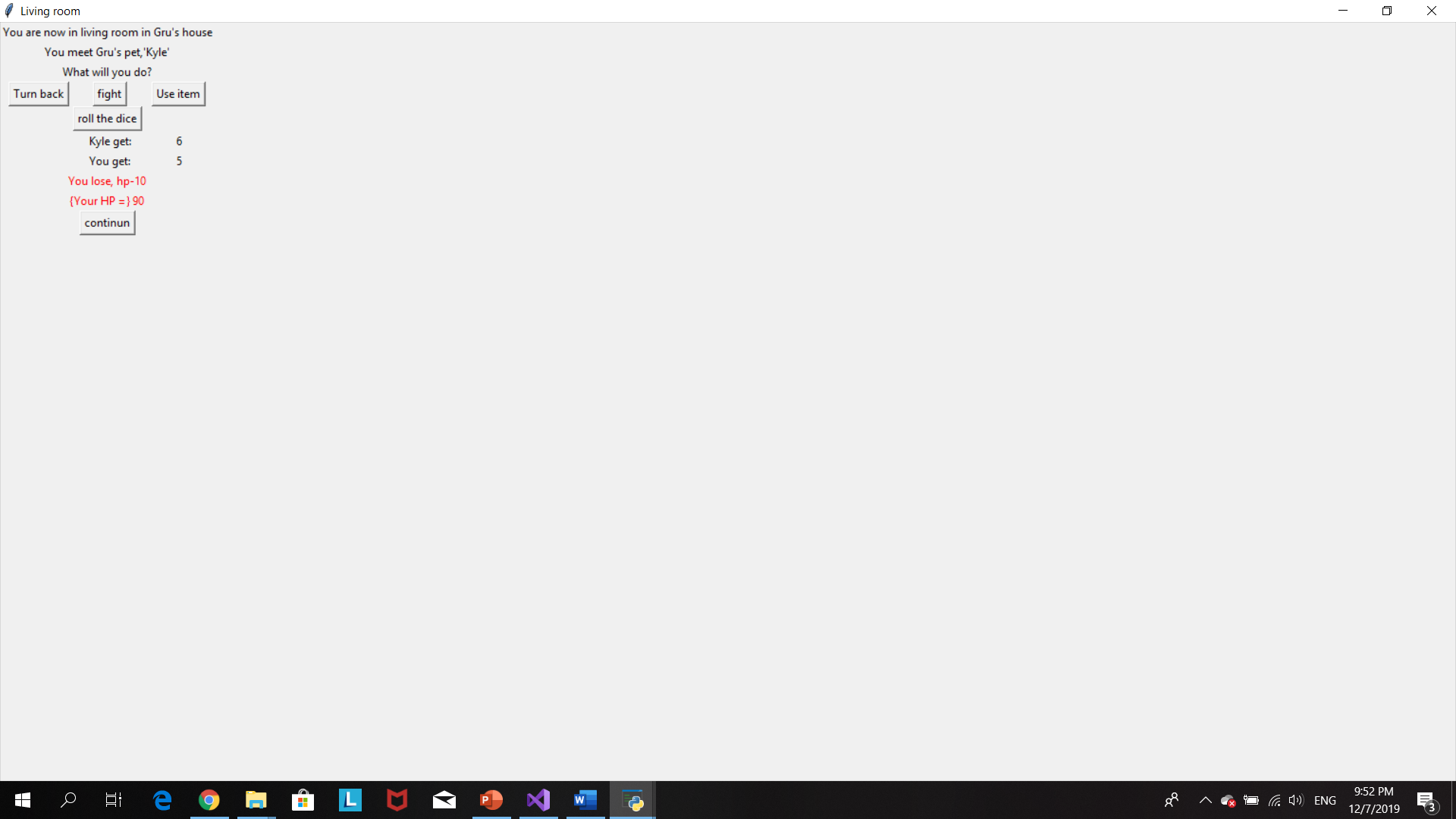


Click ‘start game’ and living room window pop up

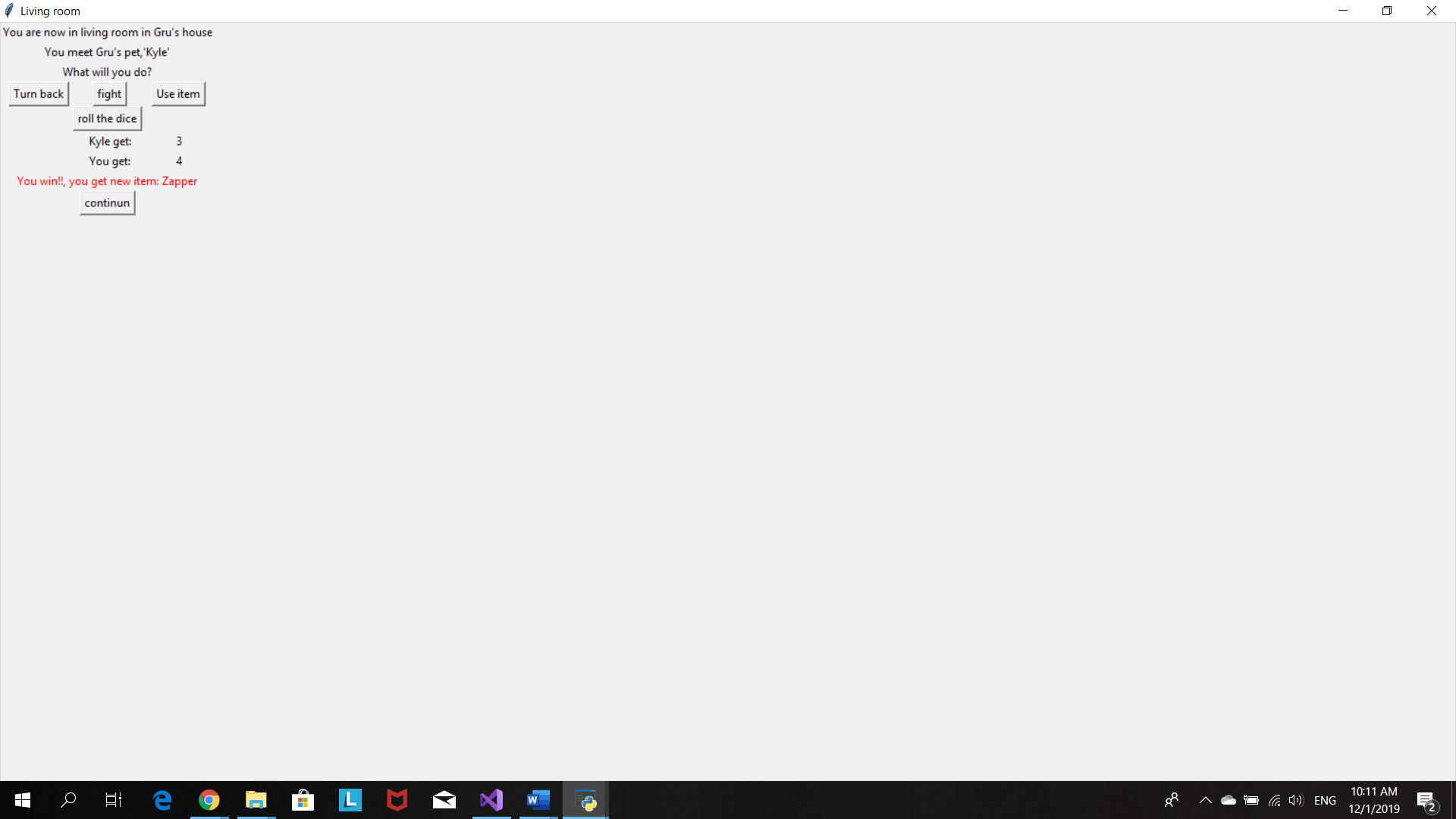


Click fight and the button roll dice will pop up then click on roll dice button the program will randon pick the number 1-6 for both player and Klye.

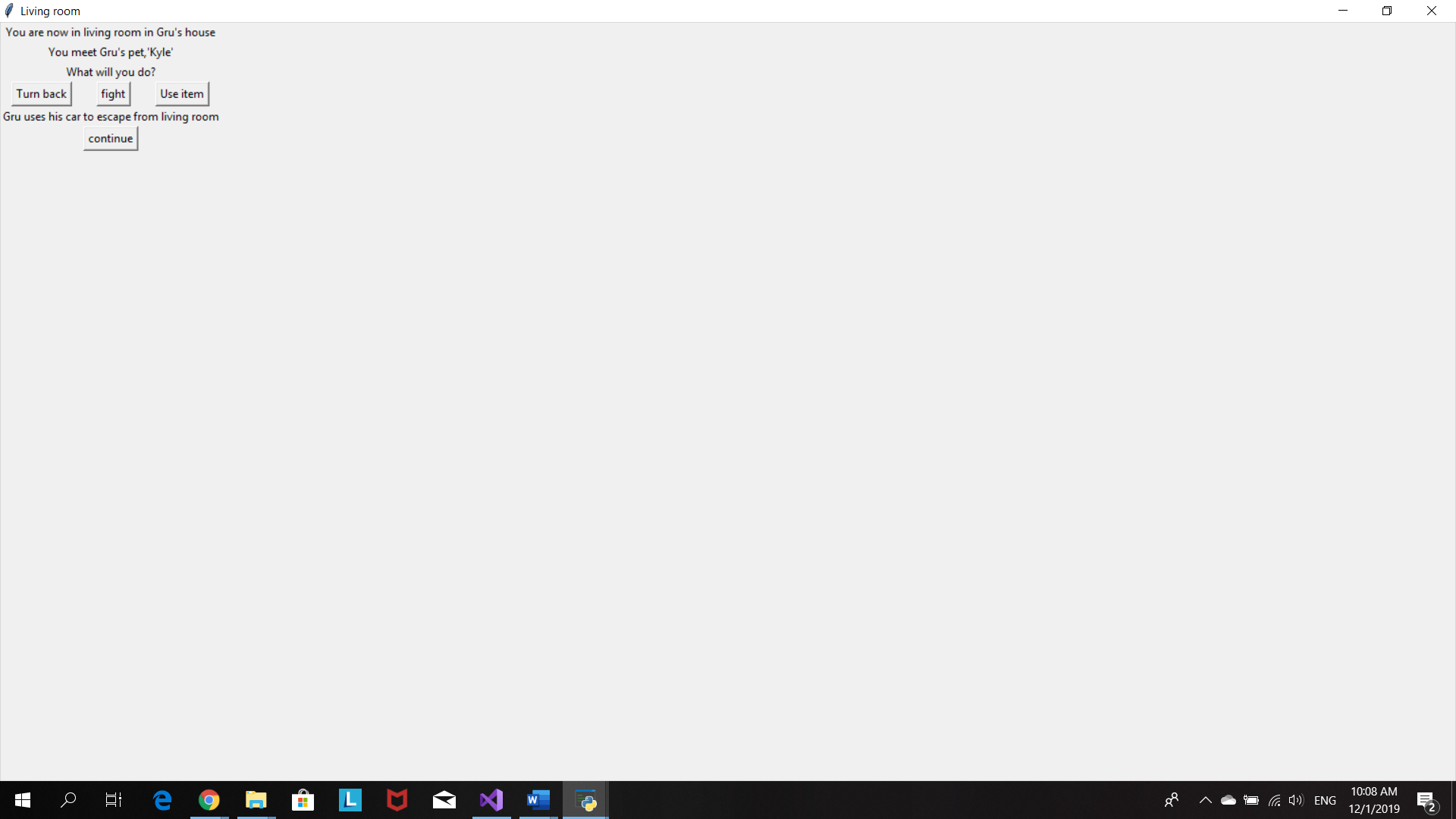
If player lost he/she will lose 10 hp



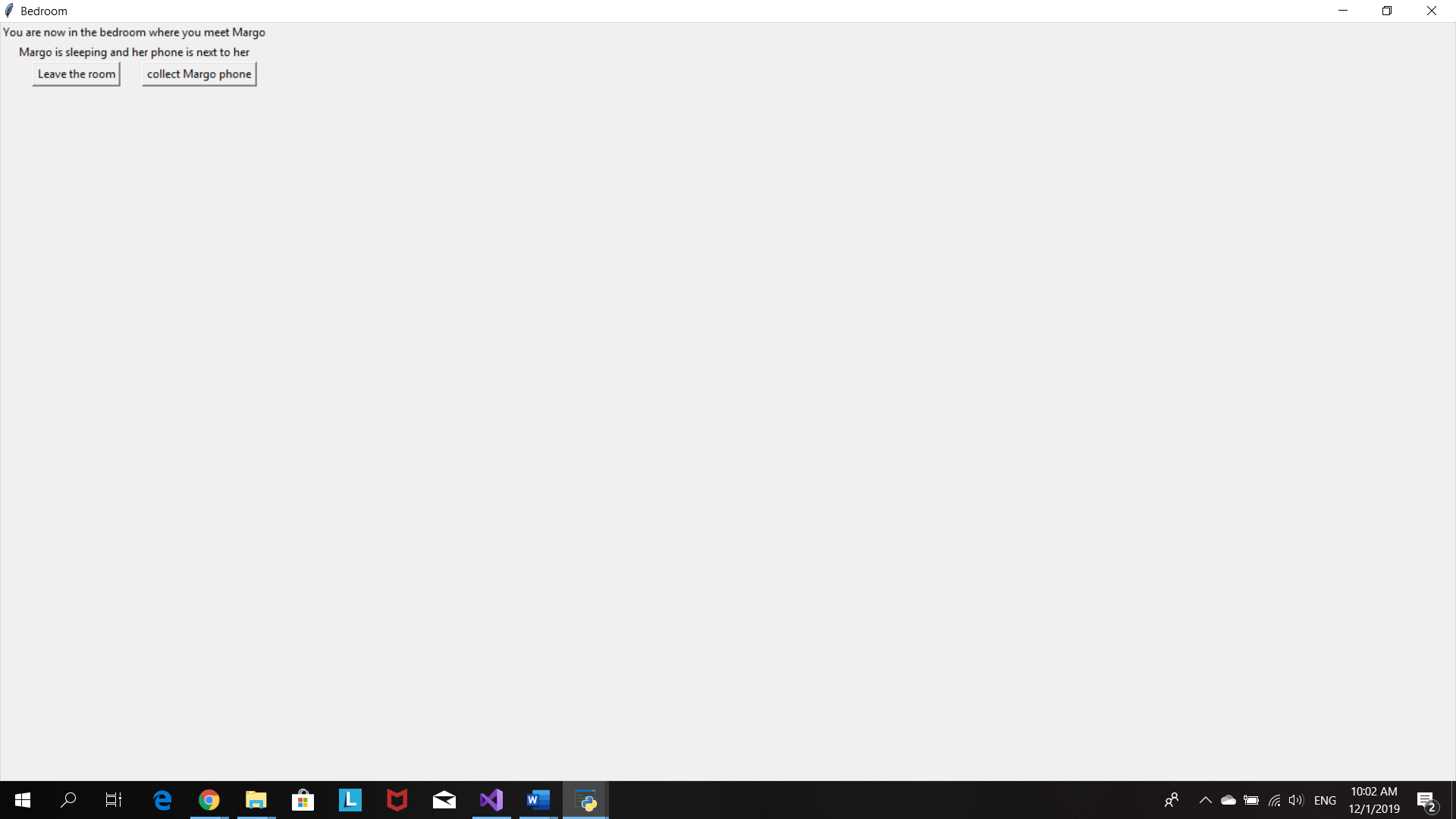
If player win they receive a Zapper item



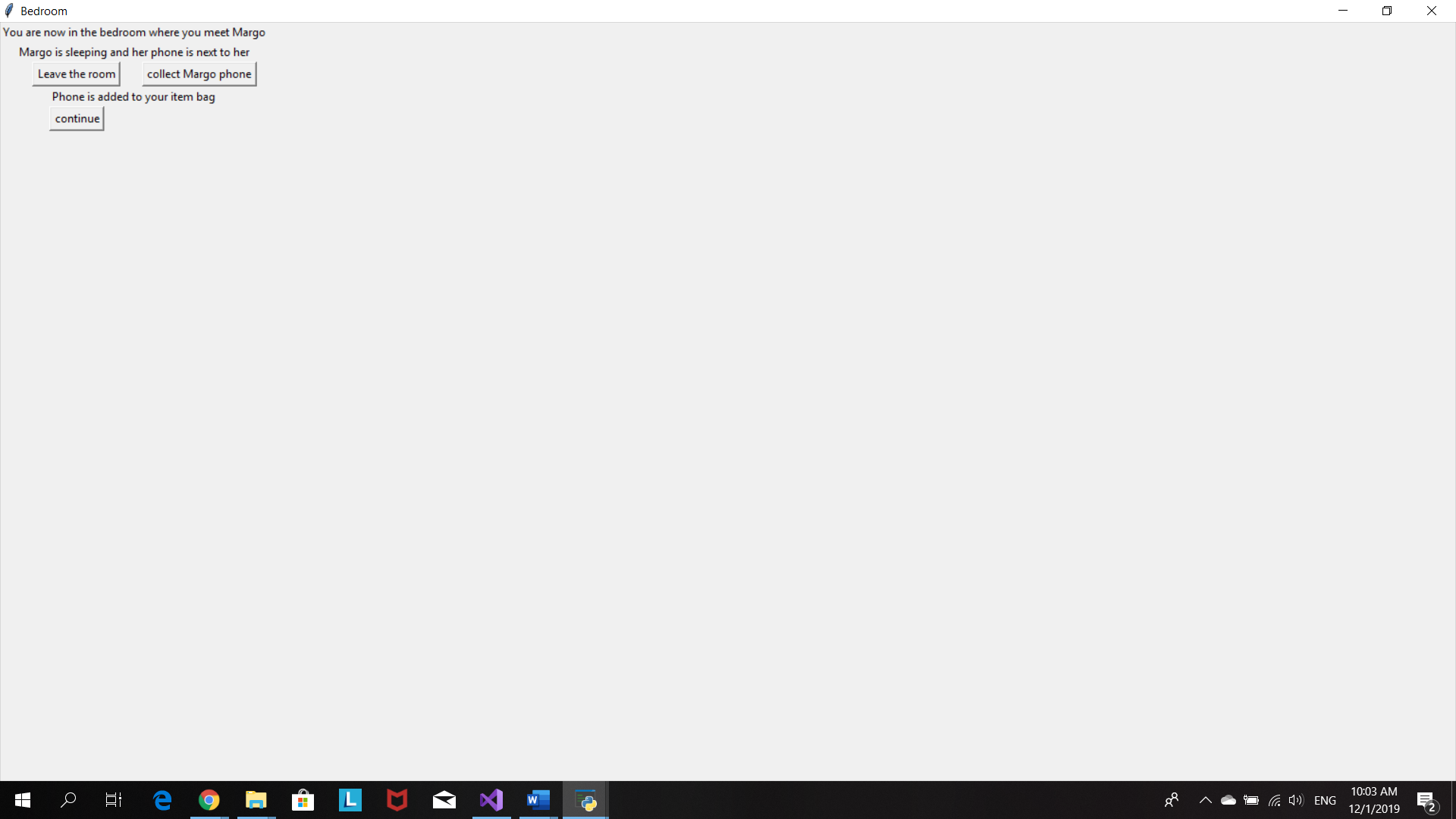
Start item will be romoved from bag when player click use item



Click continue and ‘Bedroom’ window will show up

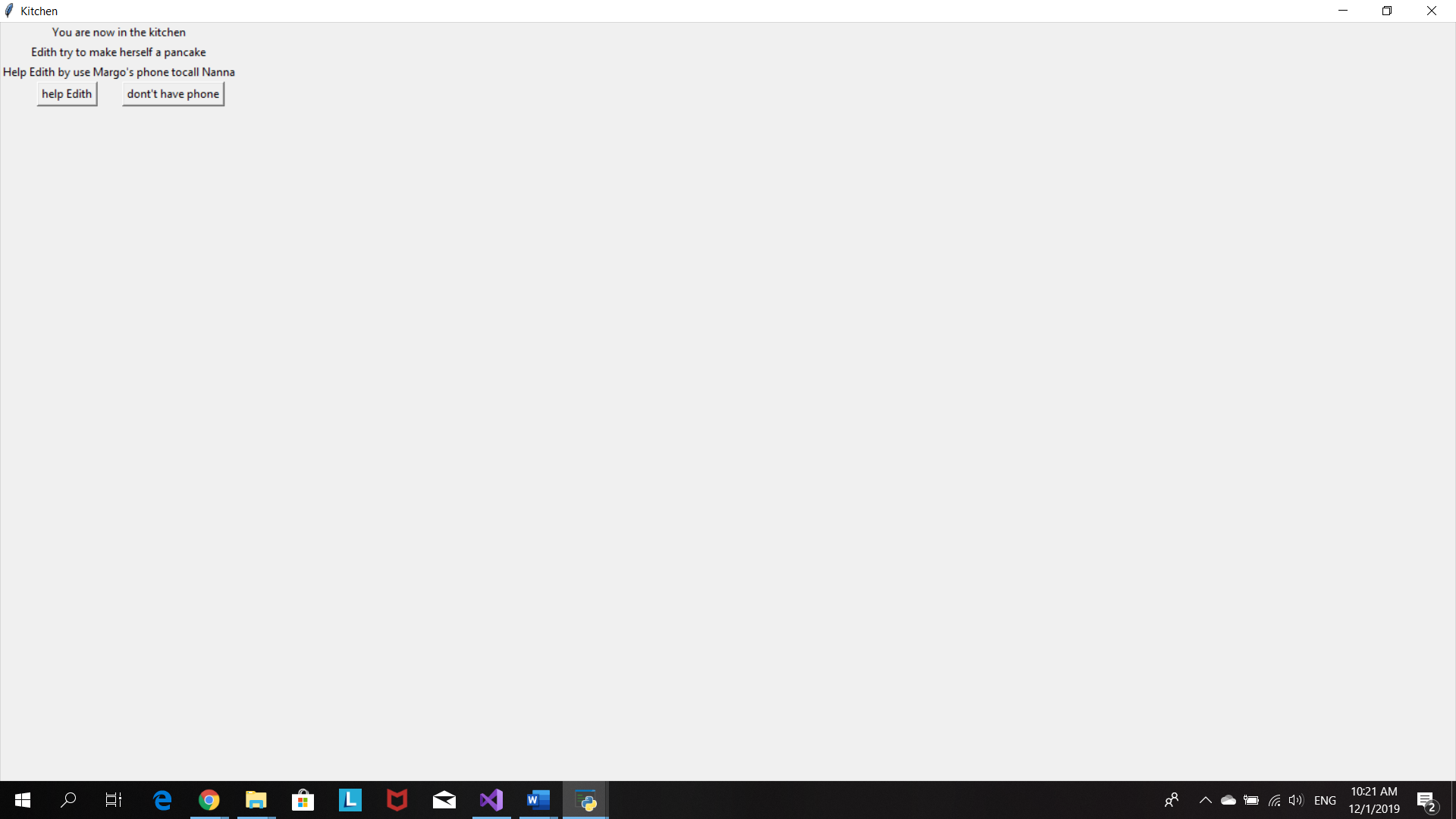


Click collect margo phone to add the item Phone into player bag.

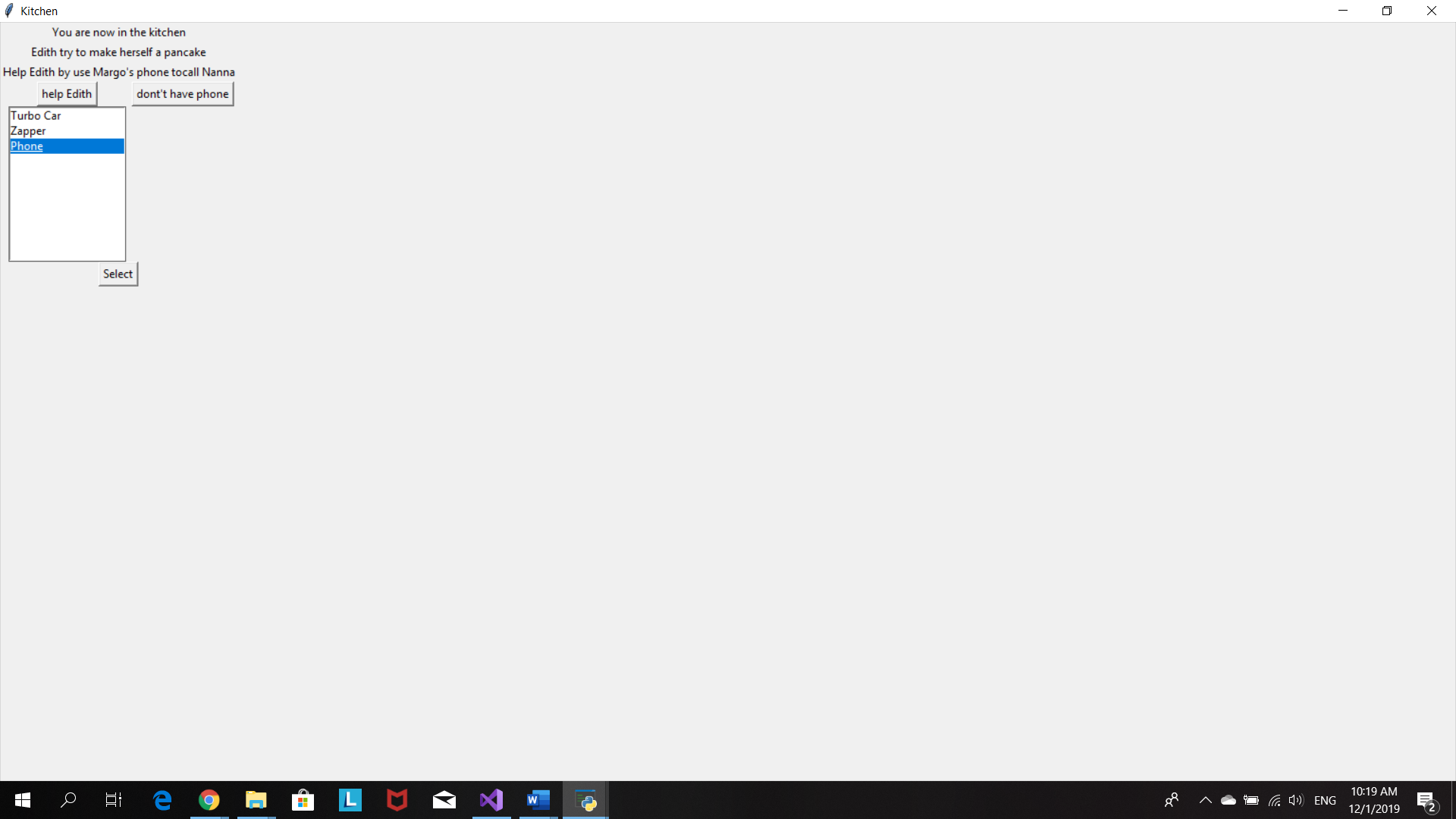


Or click leave room to go to the next scene window, kitchen .

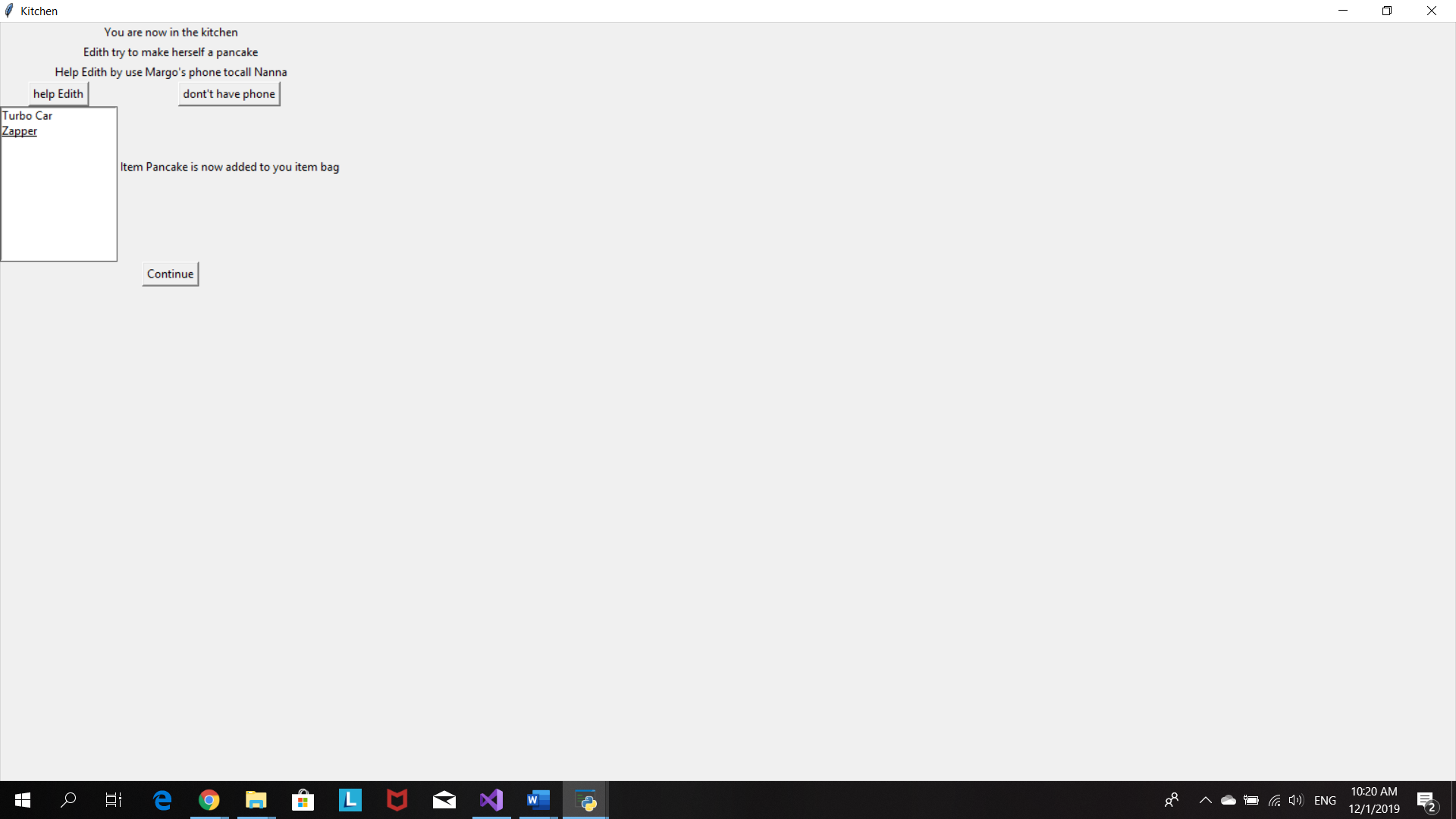
Kitchen window will pop up when player click continue after collect an Phone item as well



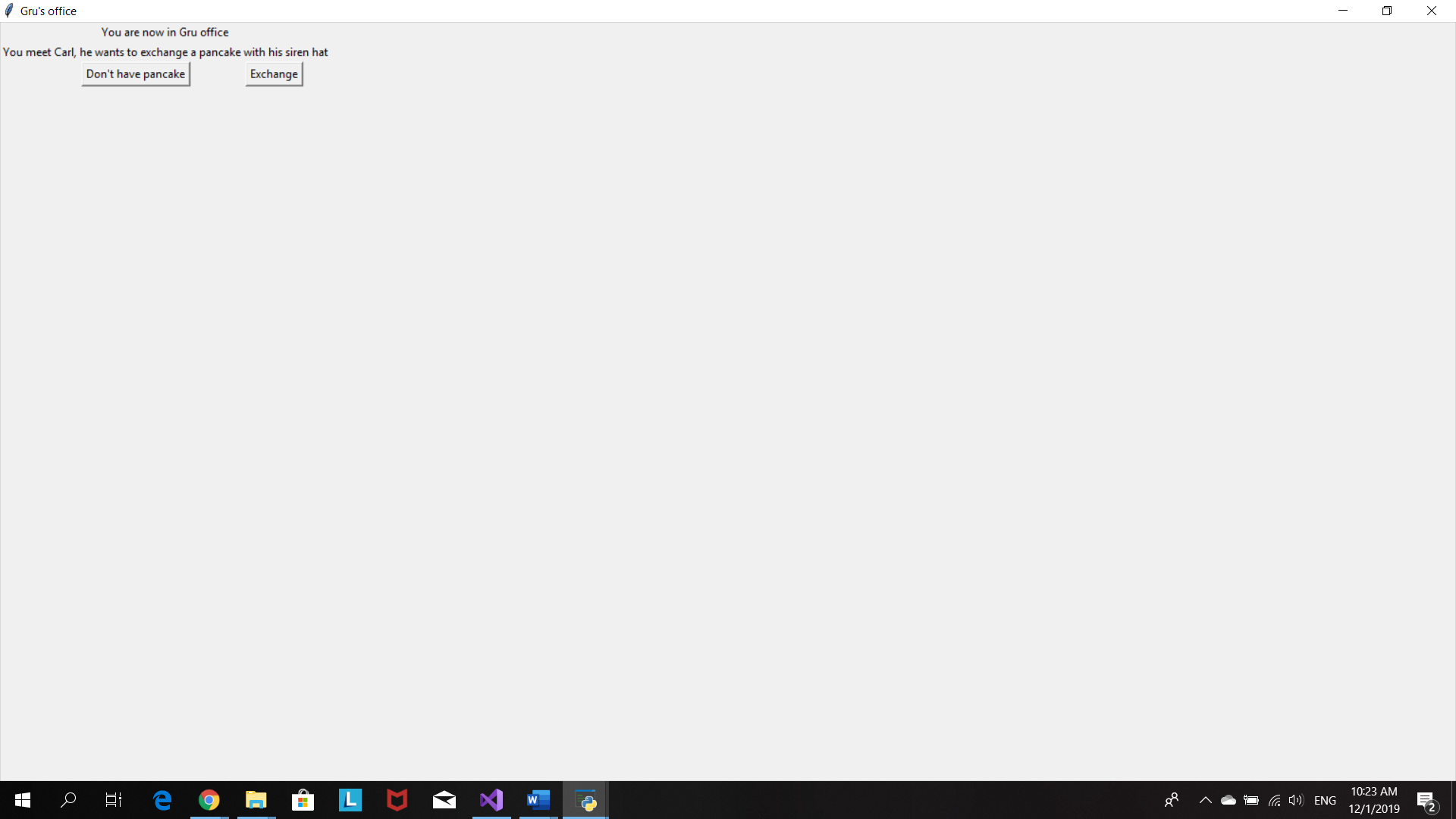
Player decied to help Edith make a pancake.Listbox pops up and all item in bag will show up for player to choose which item they want to use to help Edith



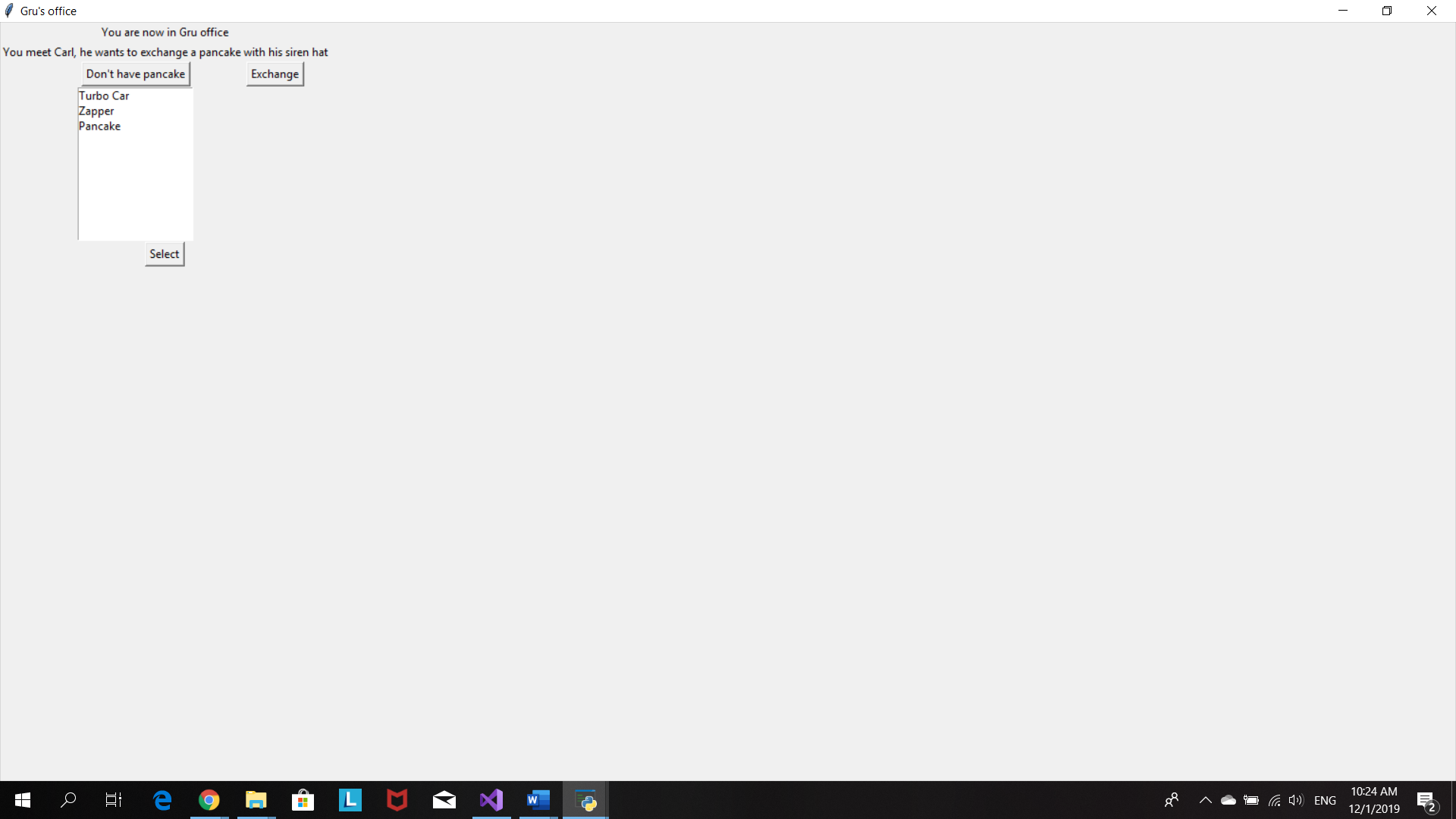
Player use phone to call Nanna. Phone is now removed from the bag and player receive new item, pancake



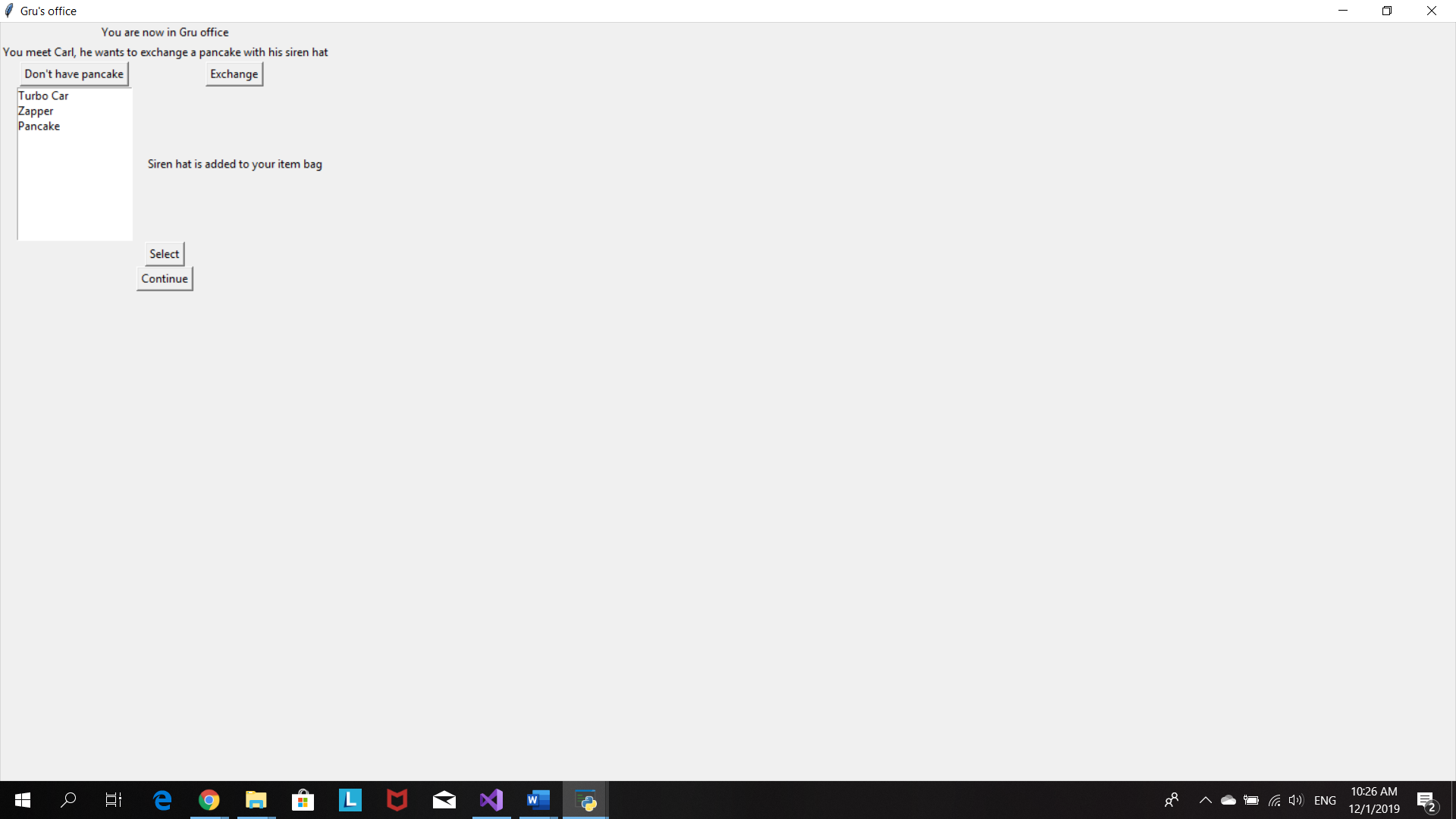
Gru’s office window will pop up when click on ‘continue’ or ‘don’t have phone’ buttons



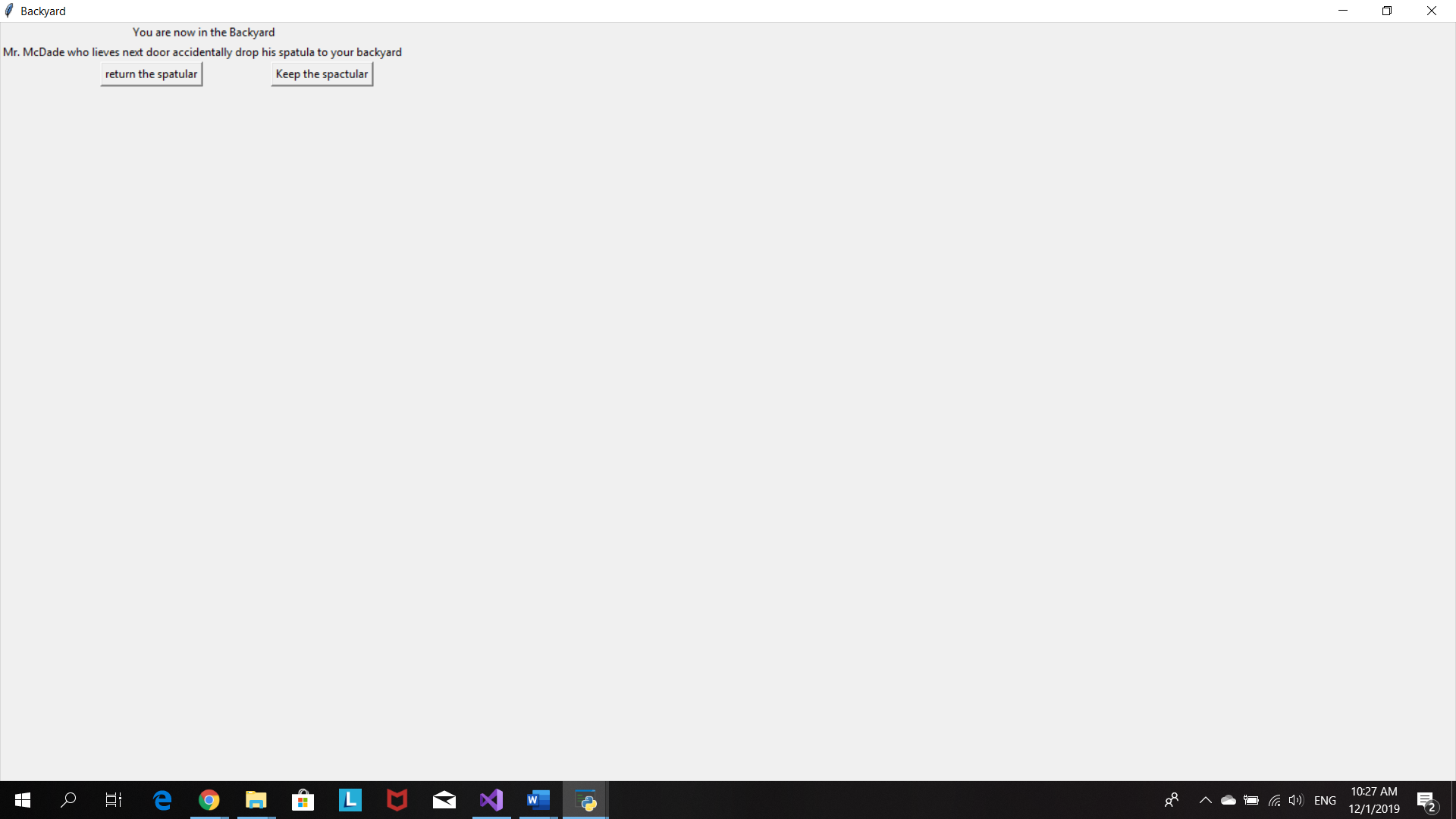
Player click on exchange button.Listbox pop up and player choose the item he/she wants to exchage with Carl’s Siren hat



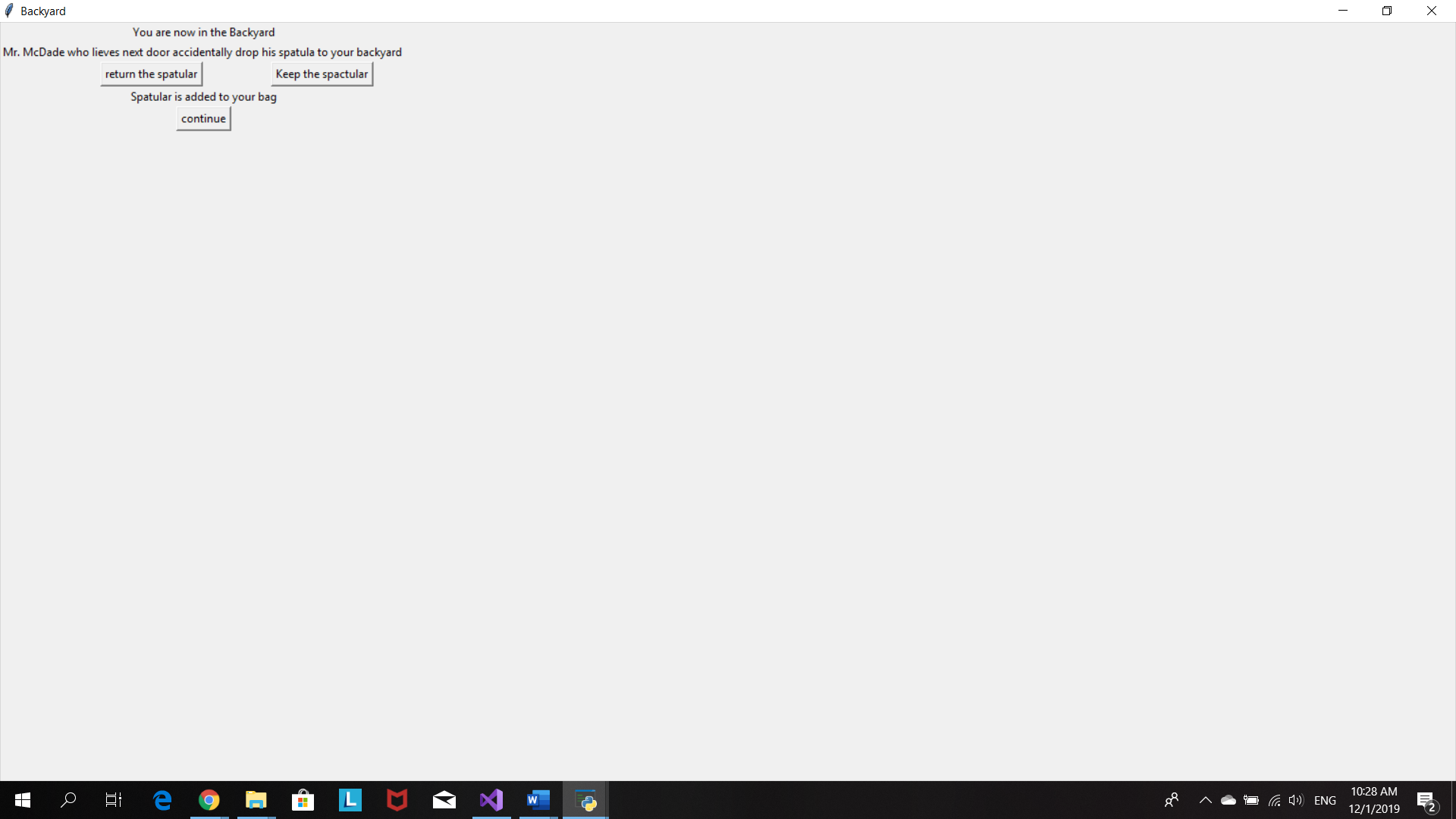
Player exchanges Pancake with Siren hat.



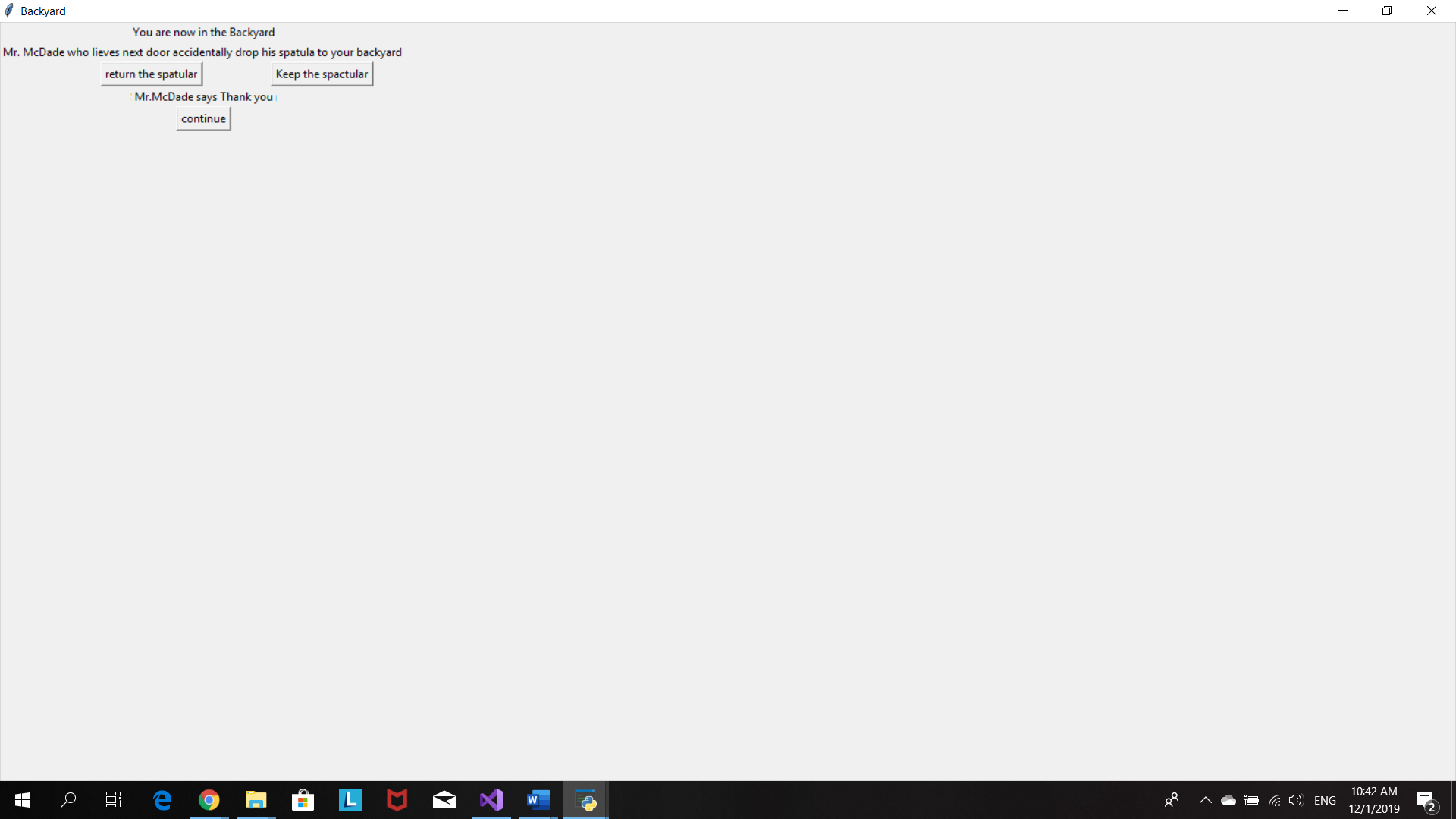
Both ‘don’t have pancake’ and ‘continue’ buttons will bring player to the Backyard window



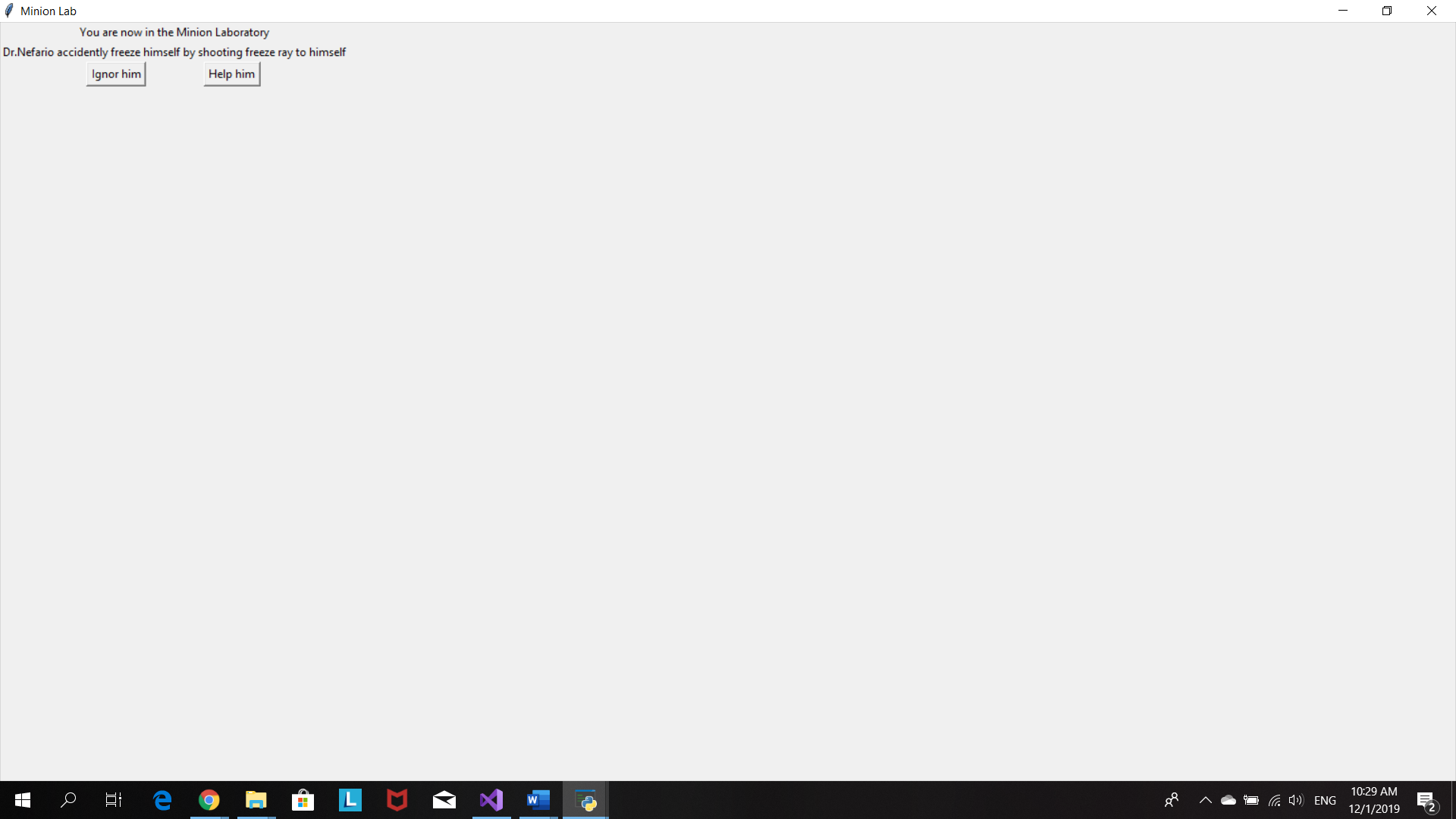
Player choose to collect Spatula



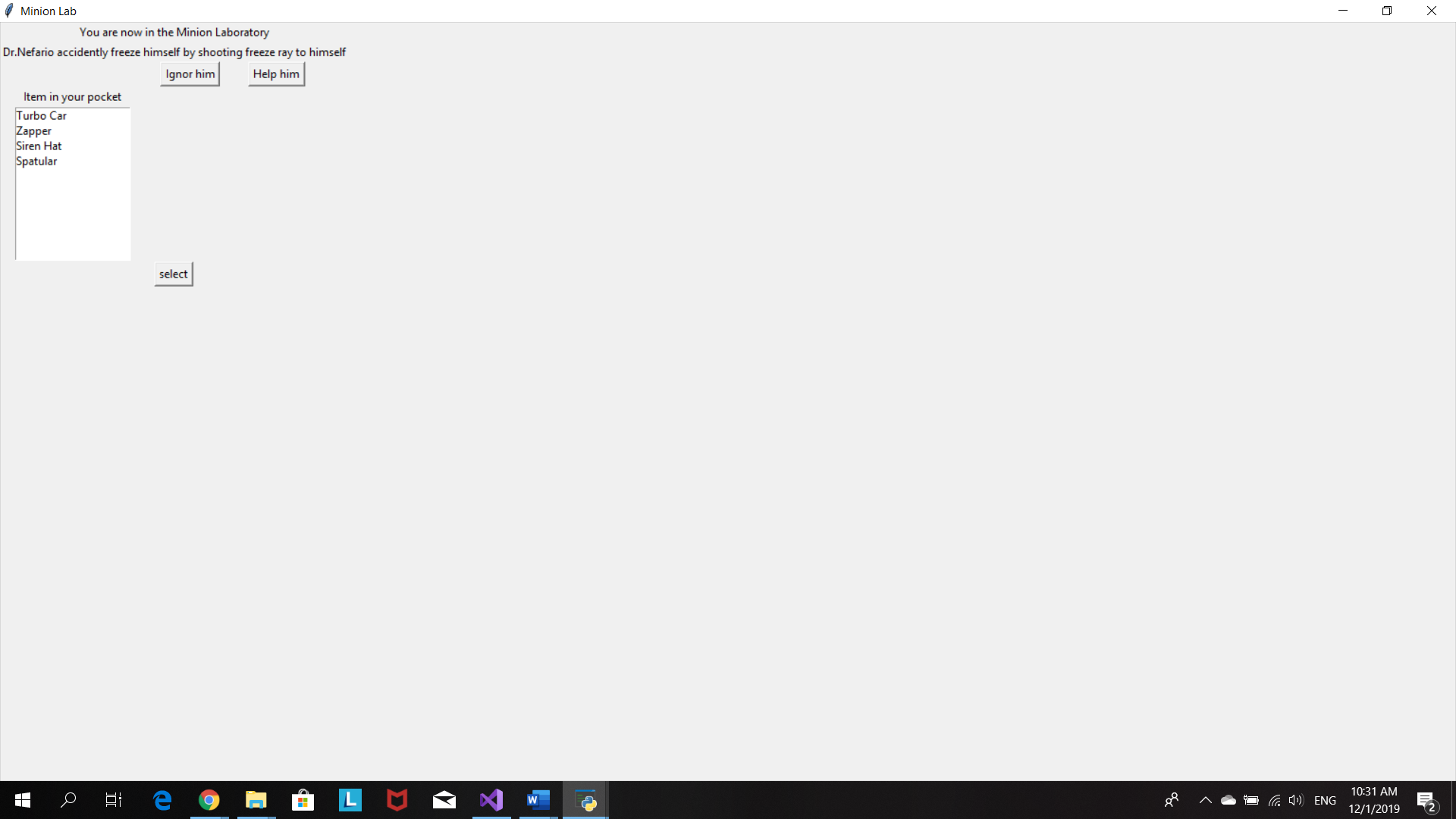
When player choose to return the Spactula



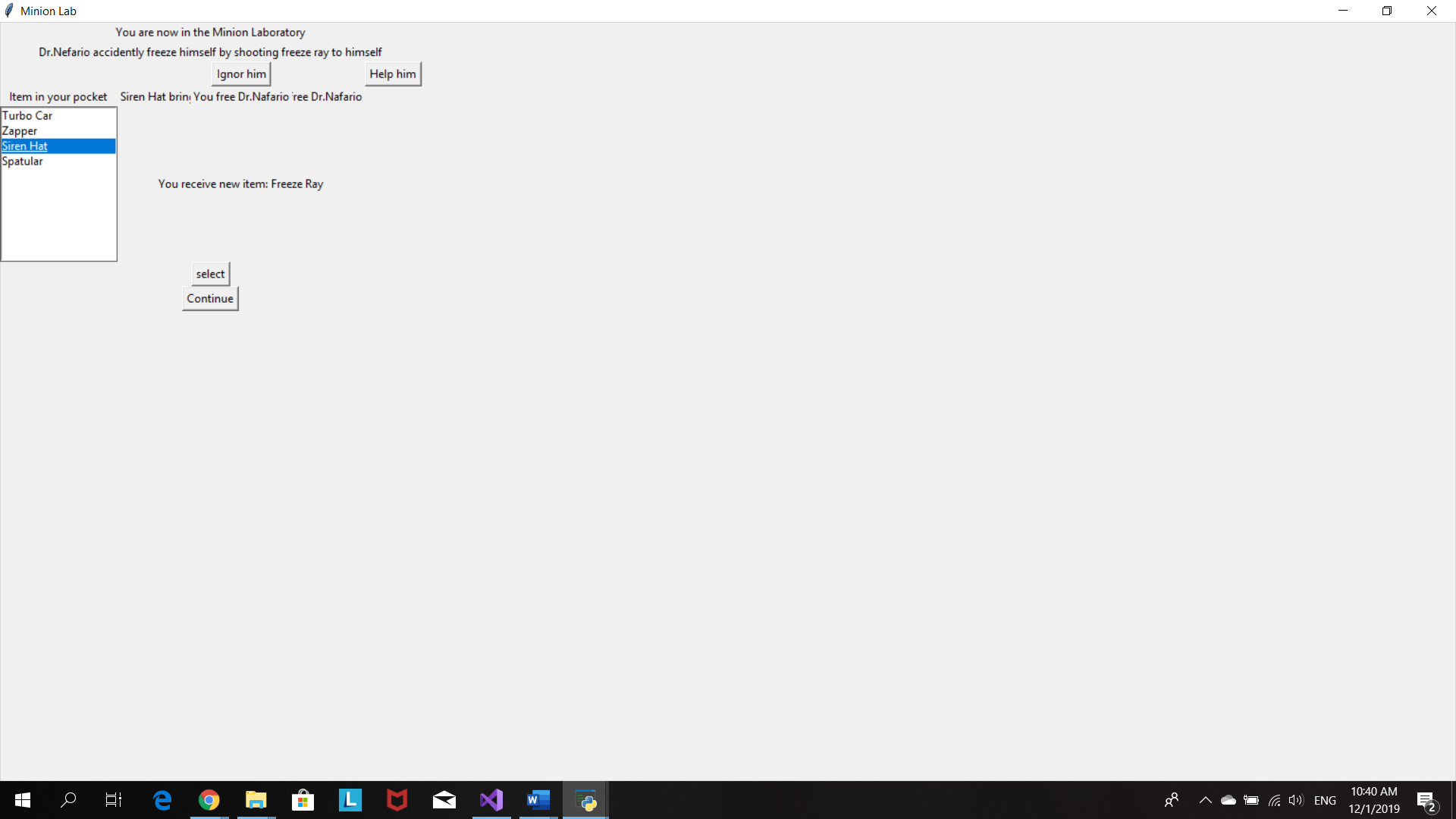
Continue button will bring player to this Minion lab window



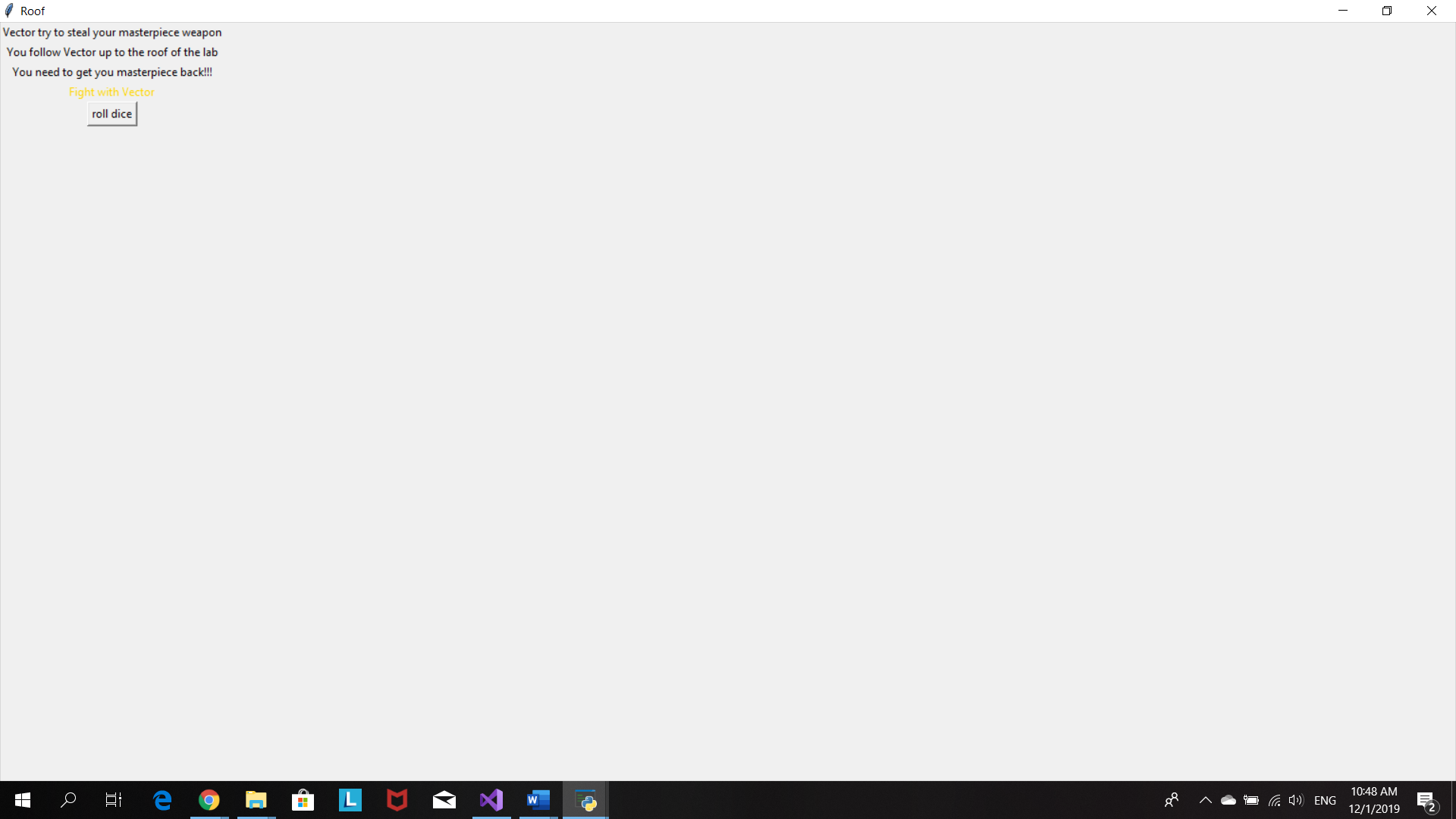
When player chooses to help Dr. Naferio, he/she will have to choose an item to help free Dr. Naferio from the ice



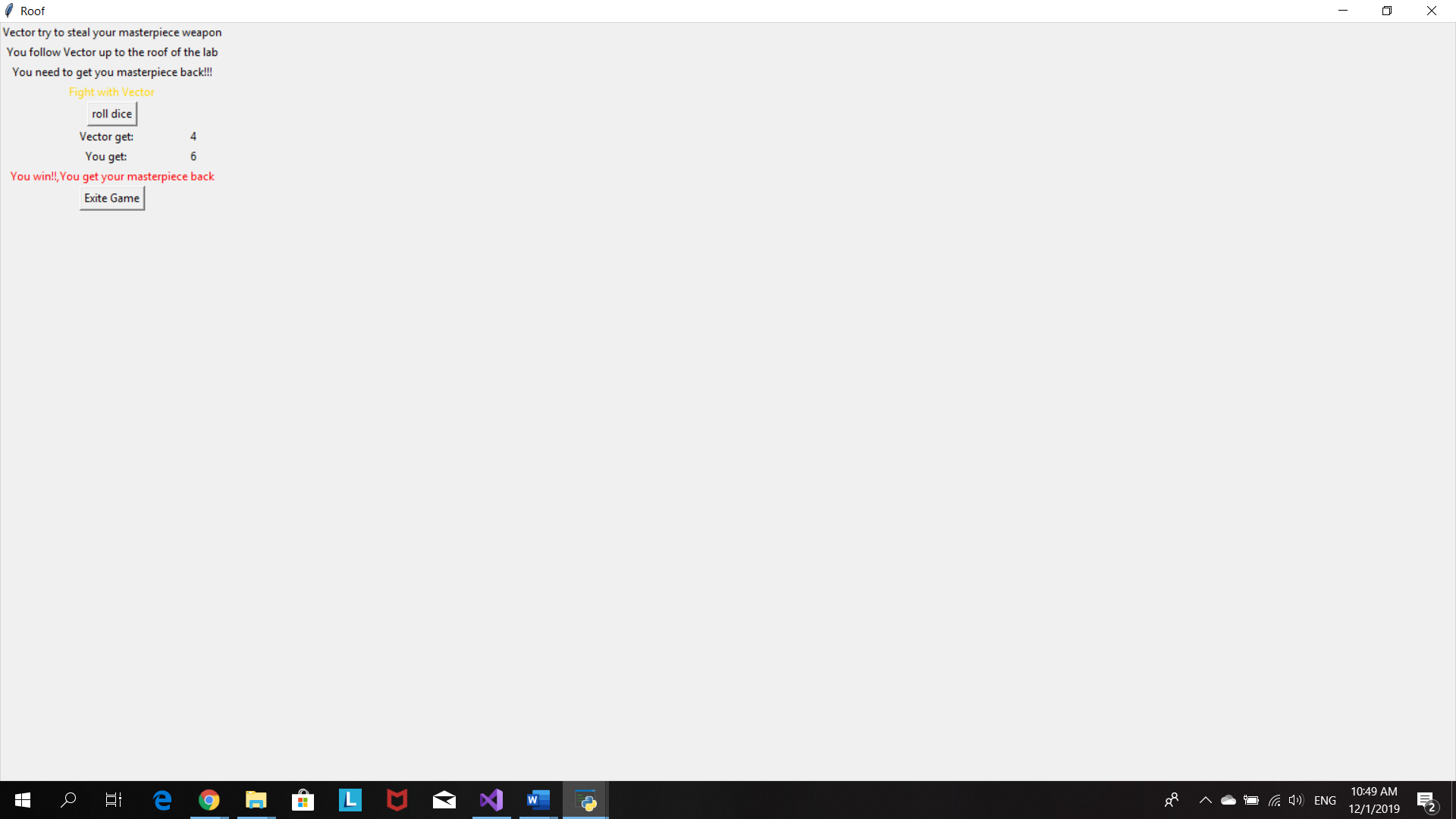
Player choose siren hat. Player free Dr. Nefario and get a new item frezze Ray



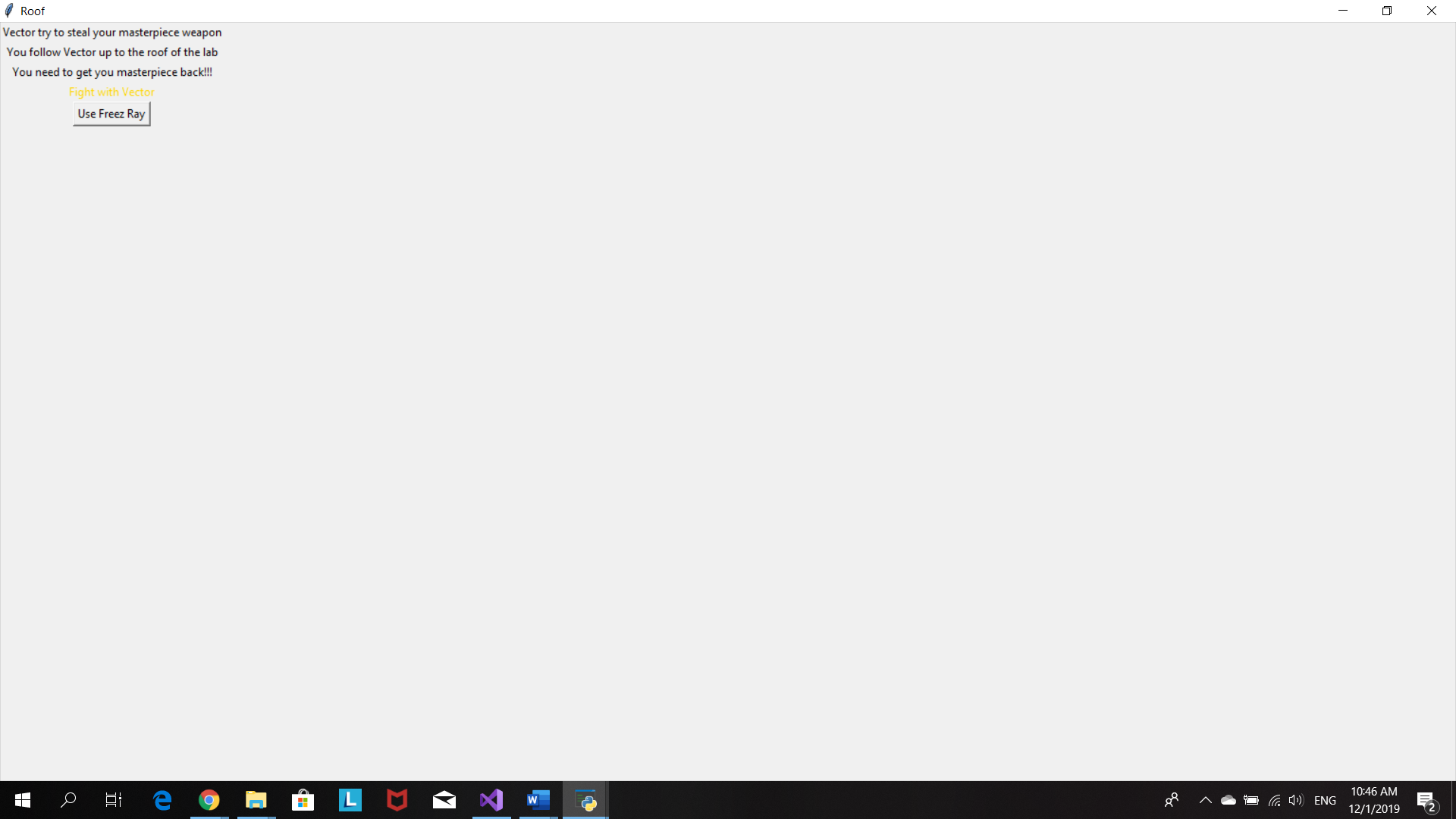
Player chooses to ignor Dr.Naferio so he/she doesn’t get the Freeze Ray.Therefore, no extra point



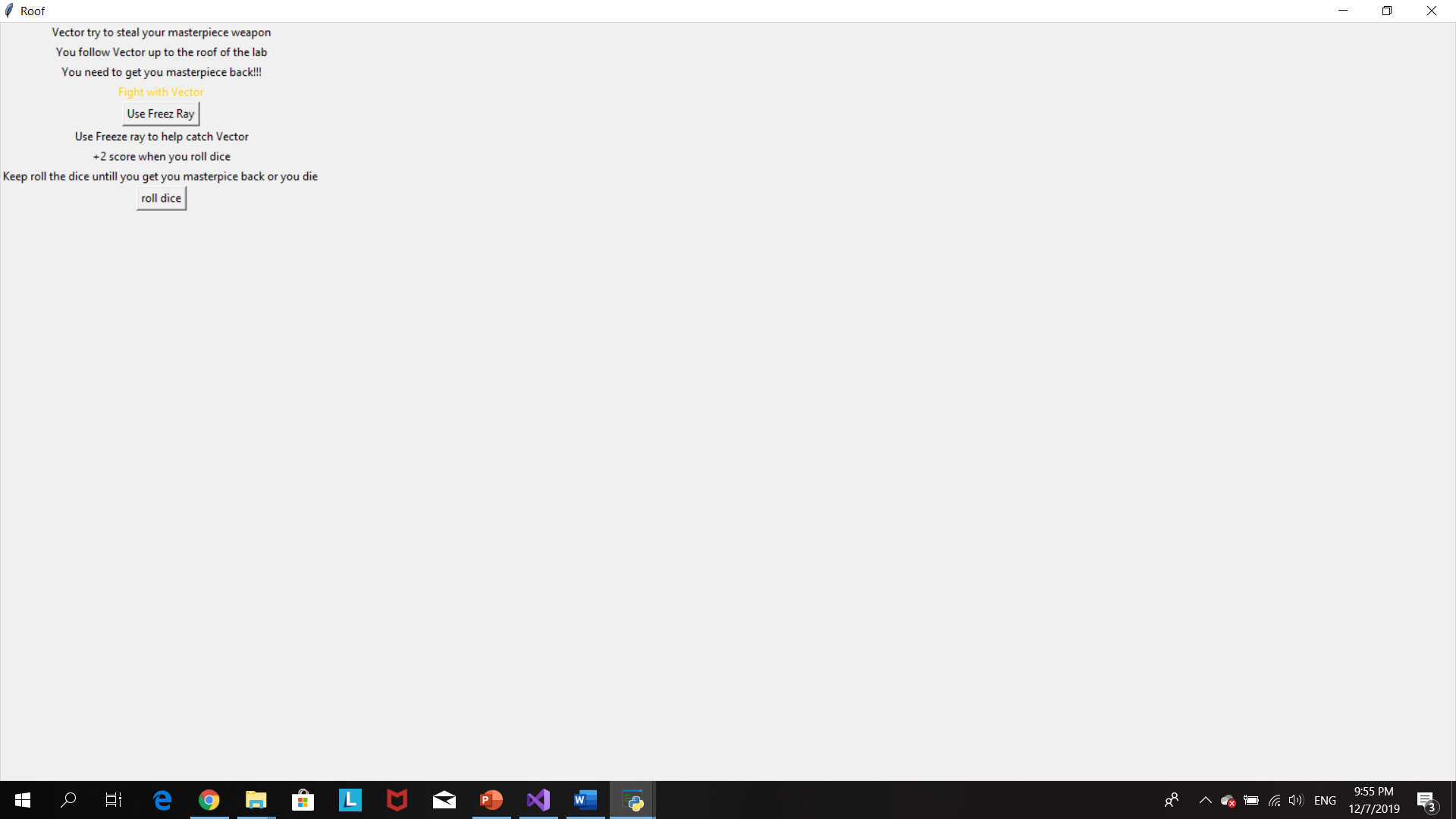
However, the player still beat Vector and have a victory



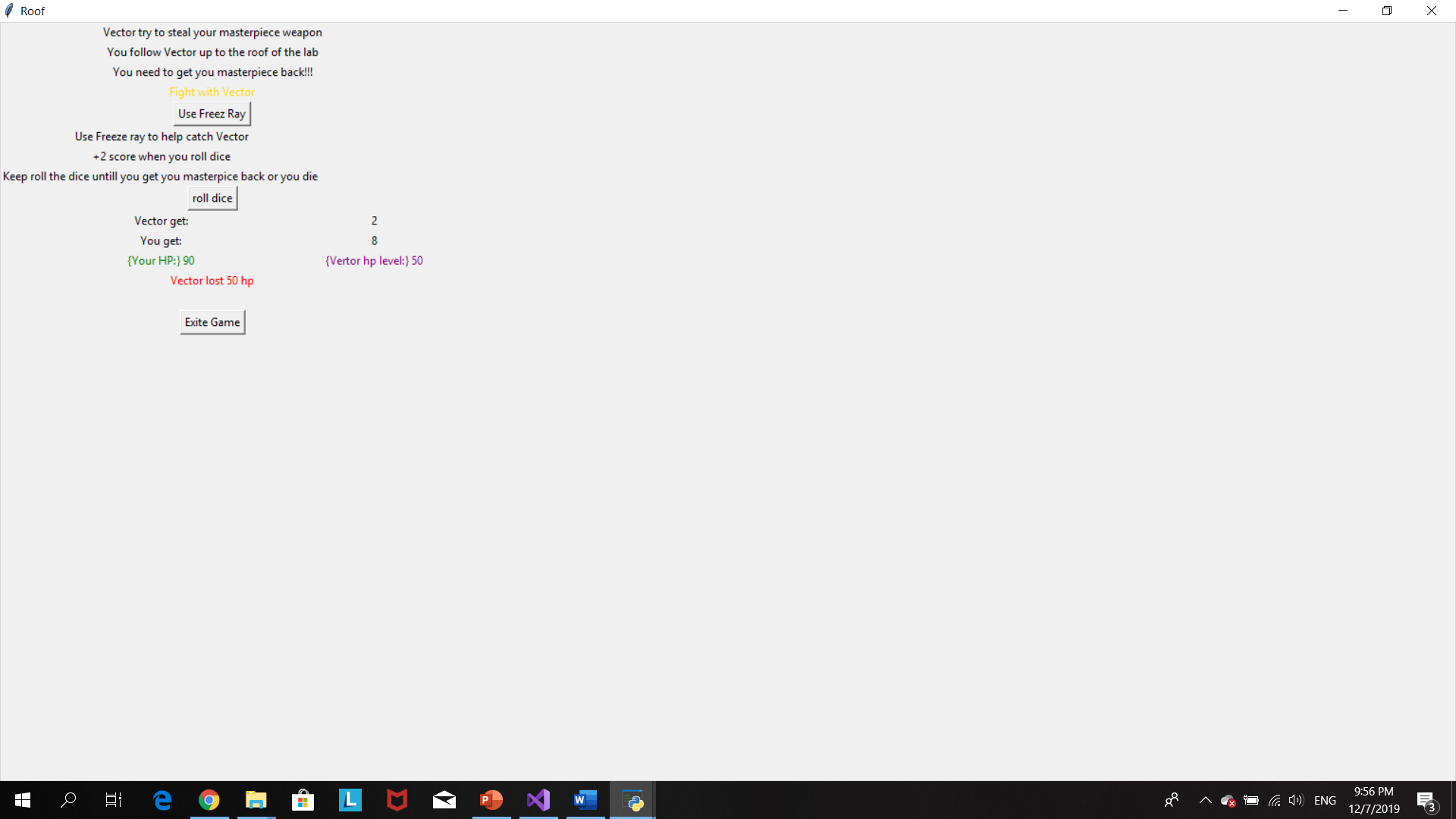
This window will pop up when user free Dr.Naferio and receive a Freeze Ray



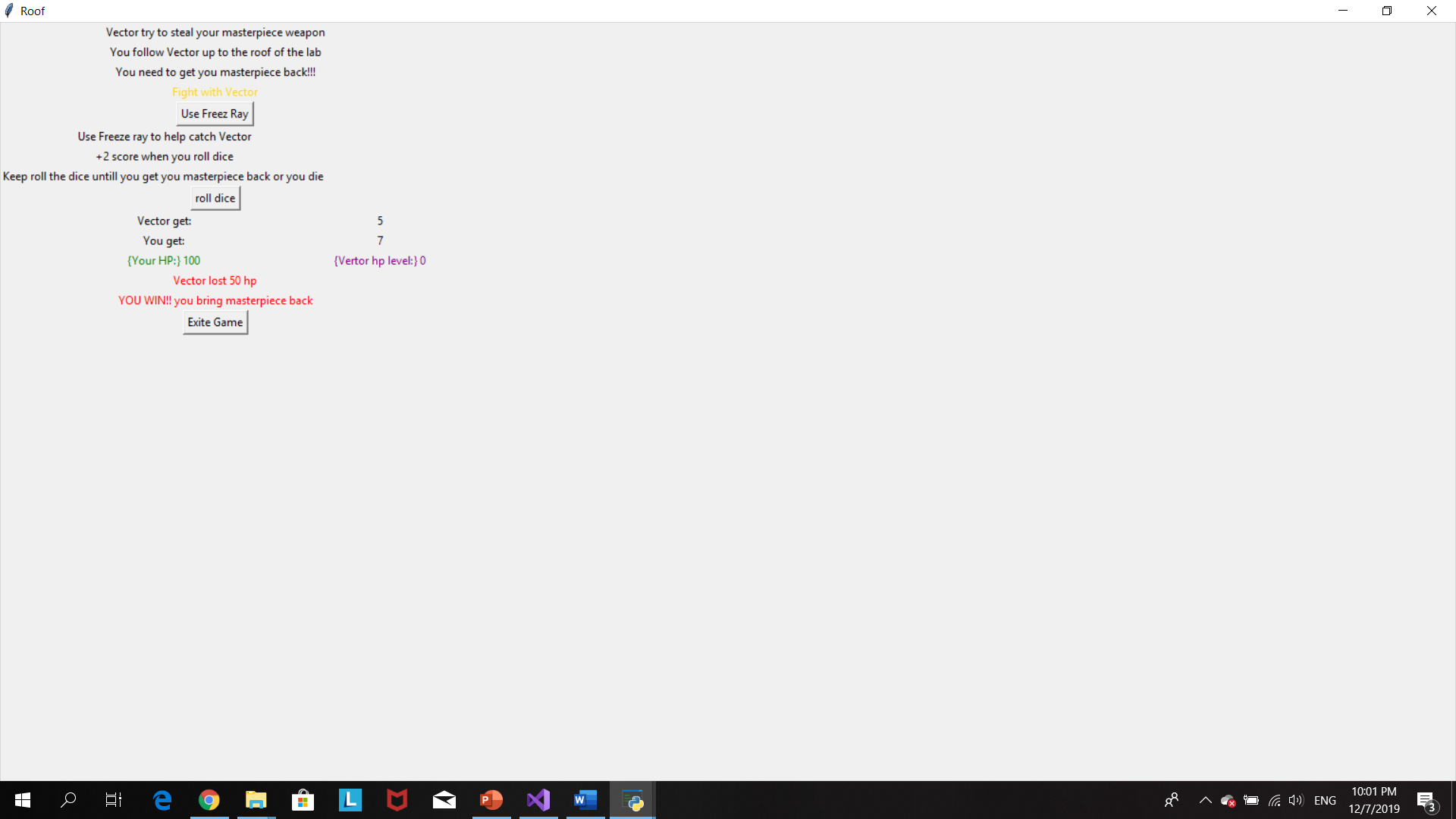
Freeze Ray +2 point fro rolling dice



Roll the dice to make an attack



Player beat Vector and get Masterpiece back



# Source Code

#Program 3: Fantasy adventure game

#The purpose of this program is to create a game inspired from the movie despicable me

#Author:Prang Kongthongluck

#Date:27/11/2019

#version:2.1

import random

from tkinter import\*

import tkinter as tk

item=[]

hplevel=100

vectorhp=100

#when player choosed Gru as his/her character

def Gruitem():

global itemlist

global item

itemlist=Listbox(root)

item.append("Turbo Car")

Label(root,text="Your HP= 100").grid(row=5,columnspan=3)

Label(root,text="Start item: Turbo car - use it to escape from anything ").grid(row=6,columnspan=3)

Button(root,text="Start Game",command=HouseGru).grid(row=7,column=1)

#window for living room scence

def HouseGru():

global house

house=Toplevel(root)

house.title("Living room")

Label(house,text="You are now in living room in Gru's house").grid(row=0,columnspan=3)

Label(house,text="You meet Gru's pet,'Kyle'").grid(row=1,columnspan=3)

Label(house,text="What will you do?").grid(row=2,columnspan=3)

Button(house,text="Turn back",command=quit).grid(row=3,column=0)

Button(house,text="fight",command=(fight)).grid(row=3,column=1)

Button(house,text="Use item",command=houseitemGru).grid(row=3,column=2)

#player use item to eascape from living room his/her start item will be removed from the item list

def houseitemGru():

Label(house,text="Gru uses his car to escape from living room").grid(row=4,columnspan=3)

item.remove("Turbo Car")

Button(house,text="continue",command=bedroom).grid(row=8,columnspan=3)

#---------------------------------------------------------------------------------------

#when player choosed Agnes as his/her character

def Agnesitem():

global item

itemlist=Listbox(root)

item.append("Unicorn")

Label(root,text="Start item: unicorn - use unicorn to trade for anything ").grid(row=4,columnspan=3)

Label(root,text="Your HP= 100").grid(row=5,columnspan=3)

Label(root,text="You have 3 lifes").grid(row=6,columnspan=3)

Button(root,text="Start Game",command=HouseAgnes).grid(row=7,column=1)

#window for living room scence

def HouseAgnes():

global house

house=Toplevel(root)

house.title("Living room")

Label(house,text="You are now in living room in Gru's house").grid(row=0,columnspan=3)

Label(house,text="You meet Gru's pet,'Kyle'").grid(row=1,columnspan=3)

Label(house,text="What will you do?").grid(row=2,columnspan=3)

Button(house,text="Turn back",command=quit).grid(row=3,column=0)

Button(house,text="fight",command=(fight)).grid(row=3,column=1)

Button(house,text="Use item",command=houseitemAgnes).grid(row=3,column=2)

#player use item to eascape from living room his/her start item will be removed from the item list

def houseitemAgnes():

Label(house,text="Agnes gives Kyle her unicorn and walk pass the living room").grid(row=4,columnspan=3)

item.remove("Unicorn")

Button(house,text="continue",command=bedroom).grid(row=8,columnspan=3)

#----------------------------------------------------------------------------------------

#when player choosed Jerry as his/her character

def Jerryitem():

global item

itemlist=Listbox(root)

item.append("Fart Gun")

Label(root,text="Start item: Fart Gun - distract attention").grid(row=4,columnspan=3)

Label(root,text="Your HP= 100").grid(row=5,columnspan=3)

Label(root,text="You have 3 lifes").grid(row=6,columnspan=3)

Button(root,text="Start Game",command=HouseJerry).grid(row=7,column=1)

#window for living room scence

def HouseJerry():

global house

house=Toplevel(root)

house.title("Living room")

Label(house,text="You are now in living room in Gru's house").grid(row=0,columnspan=3)

Label(house,text="You meet Gru's pet,'Kyle'").grid(row=1,columnspan=3)

Label(house,text="What will you do?").grid(row=2,columnspan=3)

Button(house,text="Turn back",command=quit).grid(row=3,column=0)

Button(house,text="fight",command=(fight)).grid(row=3,column=1)

Button(house,text="Use item",command=houseitemJerry).grid(row=3,column=2)

#player use item to eascape from living room his/her start item will be removed from the item list

def houseitemJerry():

Label(house,text="Jerry use Fart gun to distract Kyle and walk past the living room").grid(row=4,columnspan=3)

item.remove("Fart Gun")

Button(house,text="continue",command=bedroom).grid(row=8,columnspan=3)

#----------------------------------------------------------------------------------------

#fight with Kyle

def fight():

Button(house,text="roll the dice",command=(rollDice)).grid(row=4,columnspan=3)

#roll dice program

def rollDice():

global life

global hplevel

hplevel =100

tv\_rollKyle=StringVar()

tv\_rollplayer=StringVar()

tv\_winner=StringVar()

tv\_hp=StringVar()

#use randome for rolling dice

rollKyle = random.randint(1,6)

tv\_rollKyle.set(rollKyle)

rollplayer=random.randint(1,6)

tv\_rollplayer.set(rollplayer)

if (rollKyle>rollplayer):

winner="You lose, hp-10"

hplevel=90

heplevel=("Your HP =",hplevel)

tv\_hp.set(heplevel)

elif (rollKyle<rollplayer):

winner="You win!!, you get new item: Zapper"

item.append("Zapper")

tv\_winner.set(winner)

Label(house,text="Kyle get:").grid(row=5,column=1)

Label(house,textvariable=tv\_rollKyle).grid(row=5,column=2)

Label(house,text="You get:").grid(row=6,column=1)

Label(house,textvariable=tv\_rollplayer).grid(row=6,column=2)

Label(house,textvariable=tv\_winner,fg="red").grid(row=7,columnspan=3)

Label(house,textvariable=tv\_hp,fg="red").grid(row=8,columnspan=3)

Button(house,text="continun",command=bedroom).grid(row=9,columnspan=3)

#----------------------------------------------------------------------------------------

#bedroom scence

def bedroom():

global bed

bed=Toplevel(root)

bed.title("Bedroom")

Label(bed,text="You are now in the bedroom where you meet Margo").grid(row=0,columnspan=3)

Label(bed,text="Margo is sleeping and her phone is next to her").grid(row=1,columnspan=3)

Button(bed,text="Leave the room",anchor="e",command=Kitchen).grid(row=2,column=1)

Button(bed,text="collect Margo phone",anchor="w",command=MargoPhone).grid(row=2,column=2)

#player choose to collect Phone

def MargoPhone():

#Phone is added to the item list

item.append("Phone")

Label(bed,text="Phone is added to your item bag").grid(row=3,columnspan=3)

Button(bed,text="continue",anchor="e",command=Kitchen).grid(row=4,column=1)

#----------------------------------------------------------------------------------------

#kitchen sence

def Kitchen():

global kit

kit=Toplevel(root)

kit.title("Kitchen")

Label(kit,text="You are now in the kitchen").grid(row=0,columnspan=3)

Label(kit,text="Edith try to make herself a pancake").grid(row=1,columnspan=3)

Label(kit,text="Help Edith by use Margo's phone tocall Nanna").grid(row=2,columnspan=3)

Button(kit,text="help Edith",command=HelpEdith).grid(row=3,column=1)

Button(kit,text="dont't have phone",command=GruRoom).grid(row=3,column=2)

#display item the player have in form of listbox

def HelpEdith():

global itemlist

itemlist=Listbox(kit)

itemlist.grid(rowspan=4,column=1)

for i in item:

itemlist.insert("end",i)

Button(kit,text="Select",command=selectedItemHelpEdith).grid(rowspan =7,columnspan=3)

def selectedItemHelpEdith():

global item

#delete selected item out of th listbox

itemlist.delete(tk.ANCHOR)

#delete the selected item out from the list

try:

selection=itemlist.curselection()

value=eval(itemlist.get(selection))

ind=item.index(value)

del(item[ind])

except:

#Phone is removed out of the list

item.remove("Phone")

print("Phone is removed from the item list")

Label(kit,text="Pancake is made").grid(row=5,column=2)

Label(kit,text="Item Pancake is now added to you item bag").grid(row=5,column=2)

#Pancake is added to the item list

item.append("Pancake")

Button(kit,text="Continue",command=GruRoom).grid(row=8,columnspan=3)

#----------------------------------------------------------------------------------------

#Gru's working room scence

def GruRoom():

global gru

gru=Toplevel(root)

gru.title("Gru's office")

Label(gru,text="You are now in Gru office").grid(row=0,columnspan=3)

Label(gru,text="You meet Carl, he wants to exchange a pancake with his siren hat").grid(row=1,columnspan=3)

Button(gru,text="Exchange",command=ExchangeCarl).grid(row=2,column=2)

Button(gru,text="Don't have pancake", command=Backyard).grid(row=2,column=1)

def ExchangeCarl():

itemlist=Listbox(gru)

itemlist.grid(rowspan=3,column=1)

for i in item:

itemlist.insert("end",i)

Button(gru,text="Select",command=selectedItemExchangeCarl).grid(rowspan =6,columnspan=3)

def selectedItemExchangeCarl():

#delete item of the list when its selected by player

itemlist.delete(tk.ANCHOR)

#delete the selected item out from the list

try:

selection=itemlist.curselection()

value=eval(itemlist.get(selection))

ind=item.index(value)

del(item[ind])

except:

#pancake ir removed out of the list

item.remove("Pancake")

print("Pancake is removed from the item list")

Label(gru,text="Siren hat is added to your item bag").grid(row=4,column=2)

#Siren hat is added to the item list

item.append("Siren Hat")

Button(gru,text="Continue",command=Backyard).grid(row=20,columnspan=3)

#----------------------------------------------------------------------------------------

#backyard scence

def Backyard():

global back

back=Toplevel(root)

back.title("Backyard")

Label(back,text="You are now in the Backyard").grid(row=0,columnspan=3)

Label(back,text="Mr. McDade who lieves next door accidentally drop his spatula to your backyard ").grid(row=1,columnspan=3)

Button(back,text="return the spatular",command=returnSpatula).grid(row=2,column=1)

Button(back,text="Keep the spactular",command=keepSpatulat).grid(row=2,column=2)

def returnSpatula():

Label(back,text="Mr.McDade says Thank you").grid(row=3,columnspan=3)

Button(back,text="continue",command=Lab).grid(row=4,columnspan=3)

def keepSpatulat():

Label(back,text="Spatular is added to your bag").grid(row=3,columnspan=3)

item.append("Spatular")

Button(back,text="continue",command=Lab).grid(row=4,columnspan=3)

#----------------------------------------------------------------------------------------

#window for lab scence

def Lab():

global lab

lab=Toplevel(root)

lab.title("Minion Lab")

lab.geometry=("200\*250")

Label(lab,text="You are now in the Minion Laboratory").grid(row=0,columnspan=4)

Label(lab,text="Dr.Nefario accidently freeze himself by shooting freeze ray to himself").grid(row=1,columnspan=4)

Button(lab,text="Help him",anchor='w',command=HelpNefario).grid(row=2,column=2)

Button(lab,text="Ignor him",anchor='e',command=Roof).grid(row=2,column=1)

def HelpNefario():

global itemlist

Label(lab,text="Item in your pocket").grid(row=3,column=0)

itemlist=Listbox(lab)

itemlist.grid(row=4,column=0)

for i in item:

itemlist.insert("end",i)

Button(lab, text = "select",command=print\_HelpNafario).grid(row=6,columnspan=4)

def print\_HelpNafario():

#call a dictionary to print out the text when each item is selected

gg=Label(lab,text=aaa[itemlist.get(itemlist.curselection())]).grid(row=3,column=1)

Label(lab,text="You free Dr.Nafario",anchor='w').grid(row=4,column=1)

Label(lab,text="You receive new item: Freeze Ray").grid(row=5,column=1)

item.append("Freeze Ray")

Button(lab,text="Continue",command=RoofHelp).grid(row=7,columnspan=4)

#dictionary to display different text when each item is selected

aaa={

"Turbo Car":"Turbo Car can't melt Dr.Nafario",

"Unicorn":"Unicorn can't melt Dr.Nafario",

"Fart Gun":"Dave is anoyyed by your Fast Gun so he hits you with the Fast Gun",

"Zapper":"Zapper malt down the ice and free Dr. Nafario",

"Phone":"Turbo Car can't melt Dr.Nafario",

"Pancake":"Dave steal your pancake",

"Spatular":"Use Spatular hit the ice and free Dr.Nafario",

"Siren Hat":"Siren Hat bring all minion to help free Dr.Nafario"

}

#----------------------------------------------------------------------------------------

#this window will show up when player choose not to help Dr.Nafario

def Roof():

global roof

roof=Toplevel(root)

roof.title("Roof")

Label(roof,text="Vector try to steal your masterpiece weapon").grid(row=0,columnspan=3)

Label(roof,text="You follow Vector up to the roof of the lab").grid(row=1,columnspan=3)

Label(roof,text="You need to get you masterpiece back!!!",anchor='w').grid(row=2,columnspan=3)

Label(roof,text="Fight with Vector",anchor='w',fg="gold").grid(row=3,columnspan=3)

Button(roof,text="roll dice",command=rollDiceVector).grid(row=4,columnspan=3)

#this window will show up when the player don't have Freez ray

def rollDiceVector():

global life

global hplevel

global vectorhp

tv\_rollVector=StringVar()

tv\_rollplayer1=StringVar()

winner1=()

winner2=()

tv\_winner1=StringVar()

tv\_winner2=StringVar()

tv\_hp=StringVar()

tv\_Vhplevel=StringVar()

rollVector = random.randint(1,6)

tv\_rollVector.set(rollVector)

rollplayer1=(random.randint(1,6))

tv\_rollplayer1.set(rollplayer1)

if (rollVector>rollplayer1):

winner1="You lose, hp -30"

hplevel -=30

HP=("Your HP:",hplevel)

Vhplevel=("Vertor hp level:" ,(vectorhp))

if (rollVector<rollplayer1):

winner1="Vector lost 50 hp"

vectorhp -=50

Vhplevel=("Vertor hp level:" ,(vectorhp))

HP=("Your HP:",hplevel)

if (rollVector == rollplayer1):

winner1=(" You are tie ")

HP=("Your HP:",hplevel)

Vhplevel=("Vertor hp level:" ,(vectorhp))

if (vectorhp==0) or (vectorhp <=0):

winner2="YOU WIN!! you bring masterpiece back"

HP=("Your HP:",hplevel)

Vhplevel=("Vertor hp level:" ,(vectorhp))

if (hplevel == 0) or (hplevel <= 0):

winner2="You lose"

HP=("Your HP:",hplevel)

Vhplevel=("Vertor hp level:" ,(vectorhp))

tv\_Vhplevel.set(Vhplevel)

tv\_hp.set(HP)

tv\_winner2.set(winner2)

tv\_winner1.set(winner1)

Label(roofH,text="Vector get:").grid(row=9,column=1)

Label(roofH,textvariable=tv\_rollVector).grid(row=9,column=2)

Label(roofH,text="You get:").grid(row=10,column=1)

Label(roofH,textvariable=tv\_rollplayer1).grid(row=10,column=2)

Label(roofH,textvariable=tv\_winner1,fg="red").grid(row=12,columnspan=3)

Label(roofH,textvariable=tv\_winner2,fg="red").grid(row=13,columnspan=3)

Label(roofH,textvariable=tv\_hp,fg="green").grid(row=11,column=1)

Label(roofH,textvariable=tv\_Vhplevel,fg="purple",anchor="e").grid(row=11,column=2)

Button(roofH,text="Exite Game",command=quit).grid(row=14,columnspan=3)

#this window will show up when the player receive Freez ray by helping DR.Nafario

def RoofHelp():

global roofH

roofH=Toplevel(root)

roofH.title("Roof")

Label(roofH,text="Vector try to steal your masterpiece weapon").grid(row=0,columnspan=3)

Label(roofH,text="You follow Vector up to the roof of the lab").grid(row=1,columnspan=3)

Label(roofH,text="You need to get you masterpiece back!!!",anchor='w').grid(row=2,columnspan=3)

Label(roofH,text="Fight with Vector",anchor='w',fg="gold").grid(row=3,columnspan=3)

Button(roofH,text="Use Freez Ray",command=FreezRay).grid(row=4,columnspan=3)

def FreezRay():

Label(roofH,text="Use Freeze ray to help catch Vector",anchor='w').grid(row=5,column=1)

Label(roofH,text="+2 score when you roll dice",anchor='w').grid(row=6,column=1)

Label(roofH,text="Keep roll the dice untill you get you masterpiece back or you die ",anchor='w').grid(row=7,column=1)

Button(roofH,text="roll dice",command=rollDiceVectorH).grid(row=8,columnspan=3)

def rollDiceVectorH():

global life

global hplevel

global vectorhp

tv\_rollVector=StringVar()

tv\_rollplayer1=StringVar()

winner1=()

winner2=()

tv\_winner1=StringVar()

tv\_winner2=StringVar()

tv\_hp=StringVar()

tv\_Vhplevel=StringVar()

rollVector = random.randint(1,6)

tv\_rollVector.set(rollVector)

#+2 score for player when he use item freeze ray

rollplayer1=(random.randint(1,6))+2

tv\_rollplayer1.set(rollplayer1)

if (rollVector>rollplayer1):

winner1="You lose, hp -30"

hplevel -=30

HP=("Your HP:",hplevel)

Vhplevel=("Vertor hp level:" ,(vectorhp))

if (rollVector<rollplayer1):

winner1="Vector lost 50 hp"

vectorhp -=50

Vhplevel=("Vertor hp level:" ,(vectorhp))

HP=("Your HP:",hplevel)

if (rollVector == rollplayer1):

winner1=(" You are tie ")

HP=("Your HP:",hplevel)

Vhplevel=("Vertor hp level:" ,(vectorhp))

if (vectorhp==0) or (vectorhp <=0):

winner2="YOU WIN!! you bring masterpiece back"

HP=("Your HP:",hplevel)

Vhplevel=("Vertor hp level:" ,(vectorhp))

if (hplevel == 0) or (hplevel <= 0):

winner2="You lose"

HP=("Your HP:",hplevel)

Vhplevel=("Vertor hp level:" ,(vectorhp))

tv\_Vhplevel.set(Vhplevel)

tv\_hp.set(HP)

tv\_winner2.set(winner2)

tv\_winner1.set(winner1)

Label(roofH,text="Vector get:").grid(row=9,column=1)

Label(roofH,textvariable=tv\_rollVector).grid(row=9,column=2)

Label(roofH,text="You get:").grid(row=10,column=1)

Label(roofH,textvariable=tv\_rollplayer1).grid(row=10,column=2)

Label(roofH,textvariable=tv\_winner1,fg="red").grid(row=12,columnspan=3)

Label(roofH,textvariable=tv\_winner2,fg="red").grid(row=13,columnspan=3)

Label(roofH,textvariable=tv\_hp,fg="green").grid(row=11,column=1)

Label(roofH,textvariable=tv\_Vhplevel,fg="purple",anchor="e").grid(row=11,column=2)

Button(roofH,text="Exite Game",command=quit).grid(row=14,columnspan=3)

#Start window

root = Tk()

root.title("Select character")

Label(root,text="Welcome to Albuquerque").grid(row=0,columnspan=3)

Label(root,text="Choose youe character",anchor='w').grid(row=1,columnspan=3)

Gru=Button(root,text="Gru",fg="red",command=Gruitem).grid(row=3,column=0)

agnes=Button(root,text="Agnes",fg="pink",command=Agnesitem).grid(row=3,column=1)

jerry=Button(root,text="Jerry",fg="yellow",command=Jerryitem).grid(row=3,column=2)

root.mainloop()

# Reference

Boris, S. (2015). *Despicable Me Source, Minions assemble!*. [online] Animationsource.org. Available at: https://www.animationsource.org/despicableme/en/

Effbot.org. (2019). *The Tkinter Label Widget*. [online] Available at: http://effbot.org/tkinterbook/label.htm

Giovanni, G. (2018). *Python and Tkinter - Listbox, how to add and delete items*. [online] Youtube.com. Available at: https://www.youtube.com/watch?v=Ea1LjAyqBs4

Gundlapalli, P. (2018). *Python: tkinter print numbers based on value selected in listbox single selection.*Youtube.com. Available at: https://www.youtube.com/watch?v=GrwoIvtKUVE

JavaTpoint. (2018). *Python Tkinter Listbox - Javatpoint*. [online] Available at: https://www.javatpoint.com/python-tkinter-listbox

Klein, B. and Mitchinson, D. (2018). *GUI Programming with Python: Buttons in Tkinter*. [online] Python-course.eu. Available at: https://www.python-course.eu/tkinter\_buttons.php

Python For Beginners. (2018). *Python Game : Rolling the dice*. [online] Available at: https://www.pythonforbeginners.com/code-snippets-source-code/game-rolling-the-dice/