International Journal of Scholarly Papers for Media and Communications

ISSN 1800-7074 (Print)
ISSN 2536-5622 (On line)

Varga, M. (2022), "Contemporary and Modernized Ticket Sales Application in Virtual Reality Cinemas Made in Python", *Media Dialoques*, Vol. 15, No. 1, pp. 7-18

Contemporary and Modernized Ticket Sales Application in Virtual Reality Cinemas Made in Python

Assoc. Prof. MATIJA VARGA

North University, Koprivnica, Hrvatska, E-mail: mvarga@unin.hr

ARTICLE INFO	Received: July 21, 2021 / Revised from: August 23, 2021 Accepted: September 13, 2021 / Available online: October 15, 2020
DOI	doi.org/10.14254/1800-7074/13-4/6

ABSTRACT

The paper presents: contemporary and modernized ticket sales program created with the Python programming language, a Python programming language (application code) that created the application, an explanation of the program's main function, if-elif-else branching, a while loop, and a functional user application. The work program is presented in code view, design view and allows users to: select movies based on the selected number, select the number of tickets to be sold, select the seat you want if the seat is free, choose how many dimensions you want to watch the movie (2D, 3D, 4D, 7D, 9D or 12D), choose food or drink , choosing drinks and food based on the order number, choosing whether or not the customer is a student, and calculating the ticket price. Finally, the program will ask the user if he wants to finish the work or not. Each command used in the program is also explained in detail and the functions, loops, and corresponding branches applied.

KEYWORDS: *Python, functions, branching if-elif-else, do-while loop, application, 12D*

INTRODUCTION

An article titled “Contemporary and Modernized Ticket Sales Application in Virtual Reality Cinemas made in Python“ featured in this paper includes a program created in Python that allows users: select movies based on the selected number, choosing the number of tickets that you want to sell, seat selection you desire if the seat is free, choosing how many dimensions you want to watch the movie, choosing food or drink, selection of drinks and food given the order number, choosing if the student is a buyer or not and ticket price calculation. The goals of this paper are:

- create a program that will be functional and applicable in the so-called real (private or public) sector,
- show how to apply the program, show program options and select options,
- show the logic behind the design of the program (in code view) and using the Python programming language in which the application was created.

The Python programming language in which the application was created is detailed and explained in this textual work. Python is an interpreted language with expressive syntax that some have compared to executable pseudo code (Oliphant, 2007). Python is a general-purpose programming language, a wealth of additional information is available in several books and on Internet sites (Ibid.). The Python language is growing in popularity as a language for scientific computing, thanks to syntax, a high level standard library and several scientific packages (Guelton et al., 2015). Python attracted quickly the interest of scientists and engineers. Despite its expressive syntax and a rich collection of built-in data types (such as strings, lists, dictionaries), it became clear that, to provide the necessary framework for scientific computing, Python needed to provide an array type for numerical computing (Millman and Aivazis, 2019).

Compared to other popular programming languages in scientific computing such as C++ or MATLAB, Python is an open source scripting language of simple syntax (Sheng Bao and Zhang, 2011). Python is an object-oriented scripting language and provides a procedural programming methodology (ython$_2$). Python has been used by well-known corporations such as Google, NASA, IBM, Autodesk, etc. (Python$_1$). The paper explains: main program function, if-elif-else branching and while loop (W3SCHOOLS). The program is presented in code and design view. In the end, the program prints a form in which the user can select: finish working or not and whether he wants to complete the application process or not. Each command used in the program is also explained in detail, and loops and branches applied in the application code view. From this it is evident that the scientific method was used in the paper to analyse the scientific contents and to analyse the innovative application made in Python.

2. A TICKET SALES APPLICATION CREATED WITH THE PYTHON PROGRAMMING LANGUAGE

A ticketing application created in the Python programming language is introduced in this chapter with using the final version of the program (i. e. applications) running option „Run Module". This paper describes the Python programming language used to create an application that uses functions, branches, and loops. This paper describes the Python programming language used to create an application that uses the functions, branches, loops, capabilities and values of Boolean algebra (true and false) and imported modules such as random.

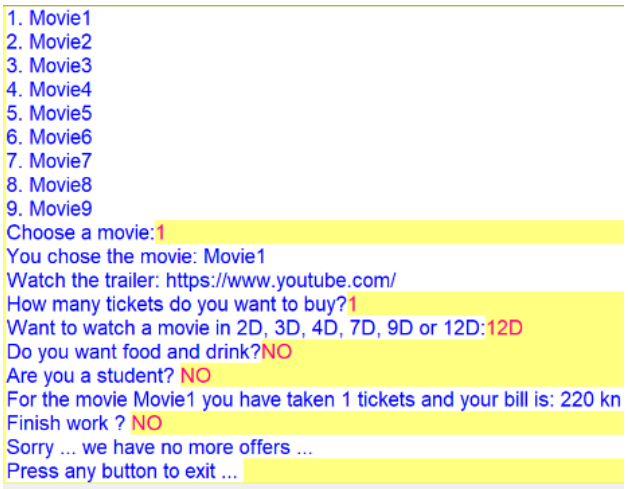

Figure 1. View Ticket Sales App

Source: Creating author work in Python programming language – Print Screen

Figure 1 (above) shows an application for the sale of tickets created in the programming language Python that has the basic functions: (1) movie selection, (2) thriller link (link to the site where the thriller can be viewed), (3) determining the number of tickets a buyer wants to buy, (4) select 2D, 3D, 4D (4D film is an immersive entertainment system that presents various physical effects with a film in order to enhance viewers' experiences (Han and Choi, 2016), 7D (7D holography projection technology that records and reconstructs a beam that propagates through space by using interferences and diffraction phenomenon and technical name of holography is wave front reconstruction. 7D holographic technologies are getting closer to augmented reality and we will enjoy virtual reality in our living rooms (Kundalakesi, M., Thenmozhi, M. (2018), 9D or 12D movie viewing modes, (5) food and drink choices, (6) the ability to state if the viewer is a student or none of the

above, (7) calculating the cost of tickets and (8) select whether or not the application user wants to quit.

2. THE PROGRAM IN PYTHON – DESIGN VIEW

Limiting Python to a statically typed subset does not hinders the expressiveness when it comes to scientific or mathematic computations, but makes it possible to use a wide variety of classical optimizations to help Python match the performances of statically compiled language. Moreover, one can use high level information to generate efficient code that would be difficult to write for the average programmer. In the end, the scientific community can benefit from a high-level language to quickly write their experiments, and still run them at decent speed (Guelton et al., 2015).

The third chapter presents the program for ticket sales made in Python in design view to the options that are used for: food and drink choices, choosing whether the customer is a student or not a student, choosing the option whether the customer wants a romantic two-person seat or does not want a romantic two-person seat.

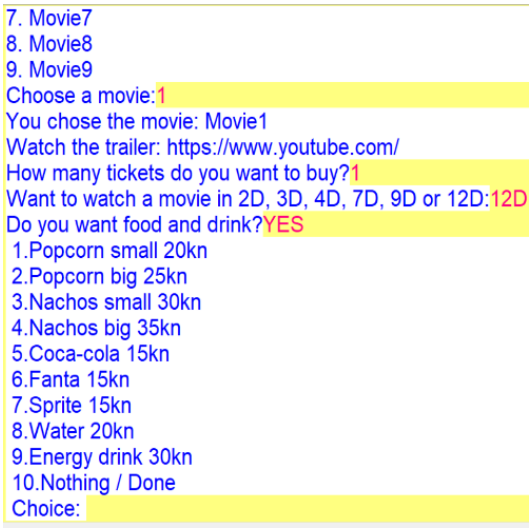

Figure 2. Food and drink choices

Source: Creating author work in Python programming language – Print Screen

Figure 2 shows the choice of food or drink type or none of the above. The application user can select the specified option by using the number in the menu. They

can choose popcorn, Nachosi, Coca-Cola, Fanta, Sprite, water, energy drinks or "almost nothing" according to the price list also shown in Figure 2.

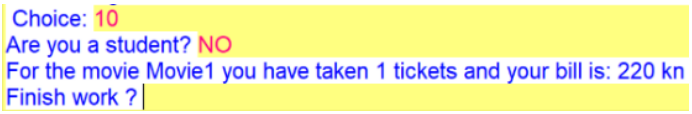

Figure 3. View options to choose if the customer is a student or not a student

Source: Creating author work in Python programming language – Print Screen

Figure 3 shows the options for choosing whether the customer is a student or not a student. If the ticket buyer is a student then the total price of the account is reduced by 50%.

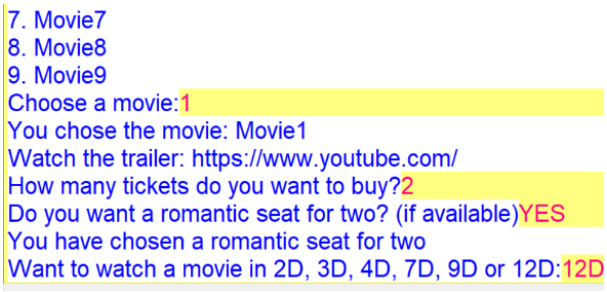

Figure 4. Show options to choose whether the customer wants a romantic seat for two or does not want a romantic seat for two

Source: Creating author work in Python programming language – Print Screen

Figure 4 shows options for numerically selecting whether the customer wants a romantic seat for two or does not want a romantic seat for two provided it is available (free). If the buyer chooses how to buy two tickets, the application gives him the opportunity to choose a romantic seat for two. If a seat is available, the system selects a romantic seat for two and then a movie projection mode. The recommendation is to choose the ability to watch a movie in 7D, 9D or 12D cinema, i.e. in virtual reality (which is certainly a novelty).

3. THE PROGRAM FOR TICKET SALES – CODE VIEW

This (fourth) chapter shows the Python programming code used to create an application that supports the ticket sales process for: 2D, 3D, 4D, 7D, 9D or 12D movie rendering modes (in virtual reality).

imported module, random ($Python_2$) randomly ejects a number, so the computer decides which is day of the week (not the user).

The Random module is imported by the import command. The Python programming language allows you to import the modules required in the program by the import command. The program itself determines the movie screening day. The user (below code) can select nine movies while randomly selecting a program to indicate on which day of the week the movie is projected. For random selection of days of the week, branching was used if-elif. Very often in life and in computer programs, the next action depends on the outcome of a question starting with “if”. This gives the possibility of branching into different types of action depending on some criterion (Linge and Langtangen, 2020). Also, branching was used in the code to select the movies we wanted to watch. The elif keyword is pythons way of saying "if the previous conditions were not true, then try this condition" (W3SCHOOLS) . The else keyword catches anything which isn't caught by the preceding conditions.

The user selects the movie, the number of tickets and the projection, if the computer randomly selects that Monday or Wednesday is the discounted price for the selected prices

imported module, random [3] randomly ejects a number, so the computer decides which is day of the week (not the user).

The user selects the movie, the number of tickets and the projection, if the computer randomly selects that Monday or Wednesday is the discounted price for the selected prices

```
import random
print("Movies:")
print("1. Movie1")
print("2. Movie2")
print("3. Movie3")
print("4. Movie4")
print("5. Movie5")
print("6. Movie6")
print("7. Movie7")
print("8. Movie8")
print("9. Movie9")
```

```
    a=random.randint(1,7)
    if a==1:
       day="Monday"
    elif a==2:
       day="Tuesday"
    elif a==3:
       day="Wednesday"
    elif a==4:
       day="Thursday"
    elif a==5:
       day="Friday"
    elif a==6:
       day="Saturday"
    elif a==7:
       day="Sunday"

    choice=int(input("Choose a movie:"))
    if choice==1:
       movie_name="Movie1"
       print("You chose the movie:",movie_name,"\nWatch the trailer:
https://www.youtube.com/")
    elif choice==2:
       movie_name="Movie2"
       print("You chose the movie:",movie_name,"\nWatch the trailer:
https://www.youtube.com/")
    elif choice==3:
       movie_name="Movie3"
       print("You chose the movie:",movie_name,"\nWatch the trailer:
https://www.youtube.com/")
    elif choice==4:
       movie_name="Movie4"
       print("You chose the movie:",movie_name,"\nWatch the trailer:
https://www.youtube.com/")
    elif choice==5:
       movie_name="Movie5"
       print("You chose the movie:",movie_name,"\nWatch the trailer:
https://www.youtube.com/")
    elif choice==6:
       movie_name="Movie6"
       print("You chose the movie:",movie_name,"\nWatch the trailer:
https://www.youtube.com/")
    elif choice==7:
       movie_name="Movie7"
```

```
        print("You chose the movie:",movie_name,"\nWatch the trailer: https://www.youtube.com/")
    elif choice==8:
        movie_name="Movie8"
        print("You chose the movie:",movie_name,"\nWatch the trailer: https://www.youtube.com/")
    elif choice==9:
        movie_name="Movie9"
        print("You chose the movie:",movie_name,"\nWatch the trailer: https://www.youtube.com/")
    else:
        print("You haven't decided on any films!")

    if choice>=1 and choice<=9:
        number_of_tickets=int(input("How many tickets do you want to buy?"))
        if number_of_tickets==2:
            ls=True
            while ls:
                love_seat=input("Do you want a romantic seat for two? (if available)")
                if love_seat=="yes" or love_seat=="YES":
                    print("You have chosen a romantic seat for two")
                    ls=False
                if love_seat=="no" or love_seat=="NO":
                    print("You have selected separate seats ")
                    ls=False
```

The branching in the code was also used to select the seat we wanted and to select the type of projection. If the customer requests two tickets, the application asks if the customer wants a romantic seat for two or separate seats. When branching "if", the logical operator "or" was used. In this code “if choice>=1 and choice<=9:”, the logical operator "and" was used. Python will parse most expressions successfully regardless of whether the types used make sense, so an expression such as ‘[put code here]’ and ‘[extra stuff]’ will parse just as well as x>0 and x % 2 == 0 (Rivers and Koedinger, 2017).

```
        choice1=True
        while choice1:
         projection=input("Want to watch a movie in 2D, 3D, 4D, 7D, 9D or 12D:")
         if projection=="2D" or projection=="2d":
            price=30
            choice1=False
         elif projection=="3D" or projection=="3d":
            price=50
```

```
            choice1=False
        elif projection=="4D" or projection=="4d":
            price=100
            choice1=False
        elif projection=="7D" or projection=="7d":
            price=120
            choice1=False
        elif projection=="9D" or projection=="9d":
            price=160
            choice1=False
        elif projection=="12D" or projection=="12d":
            price=220
            choice1=False
        else:
            print("You have not selected a projection, dial again or leave")

    if day=="Monday":
            print("Today, all tickets are $ 25 because it's Monday")
            price=25
    elif day=="Wednesday":
        print("Today's action is Wednesday and tickets are down 15%")
        price=price*0.85
    else:
        pass

    price_food=int
    price_food=0
    choice1=True
    while choice1:
        food=input("Do you want food and drink?")

        if food=="yes" or food=="YES":
            choice1=False
            jos_food="yes"
            while jos_food=="yes":
                choice_food=input(" 1.Cooks small 20kn \n 2.Cooks big 25kn \n
3.Nachos small 30kn \n 4.Nachos big 35kn \n 5.Coca-cola 15kn \n 6.Fanta 15kn \n
7.Switch 15kn \n 8. Water 20kn \n 9.Energy drink 30kn \n 10.Nothing/Done\n
Choice:")
```

The program uses a while loop that allows you to select food and drink. If the user wants food and drink, he selects yes and after selecting an option, an offer listing

the food and drink entered will be printed. With the while loop we can execute a set of statements as long as a condition is true (W3SCHOOLS).

```
if choice_food=="1":
            price_food=price_food+20

        elif choice_food=="2":
            price_food=price_food+25

        elif choice_food=="3":
            price_food=price_food+30

        elif choice_food=="4":
            price_food=price_food+35

        elif choice_food=="5":
            price_food=price_food+15

        elif choice_food=="6":
            price_food=price_food+15

        elif choice_food=="7":
            price_food=price_food+15

        elif choice_food=="8":
            price_food=price_food+20

        elif choice_food=="9":
            price_food=price_food+30

        elif choice_food=="10":
            jos_food="ne"

        else: print("please select choice 1-10")

    elif food=="no" or food=="NO":
        choice1=False

    else: print ("please answer yes or no")

account=price*number_of_tickets+price_food
st=True
while st:
```

```
    student=input("Are you a student? ")
    if student=="yes" or student=="YES":
        st=False
        account=account*0.5
    if student=="no" or student=="NO":
        st=False

print("For the movie " + movie_name+ " you have taken " + str(number_of_tickets)+
" tickets and your bill is: " + str(account)+" kn ")
finish=input ("Finish work ? ")
if finish=="yes" or finish=="YES":
  exit()
else: print ("Sorry ... we have no more offers ...")
end=input ("Press any button to exit ... ")
```

Python can be used in exactly this way, but its unique features offer an environment that makes it a better choice for scientists and engineers seeking a high-level language for writing scientific applications. Python excels as a platform for scientific computing (Oliphant, 2007).

CONCLUSION

Based on the research in this paper on "*Contemporary and Modernized Ticket Sales Application in Virtual Reality Cinemas Made in Python*" we conclude that work (application) is of utmost importance in supporting the ticket sales process for 2D, 3D, 4D, 7D or 12D cinema projection. They are especially interesting 7D and 12D cinema projections as virtual reality. Programming is known to be used in almost every area of the profession as a command-writing process in a particular programming language as well as in all fields of science to build applications that will enhance processes or individual modules within institutions or companies (private or public corporations) no matter what activity they are engaged in. The contribution of the work can be seen by displaying the final application to support the ticket sales process for 2D, 3D, 4D, 7D or 12D cinema projection and the sale of certain foods that can be consumed in the most elite cinema. Because of its strength and flexibility, Python is the ideal choice for problem solving across domains.

REFERENCES

Guelton, S. et al. (2015), “Pythran: enabling static optimization of scientific Pyt-hon programs”, *Computational Science & Discovery*, Vol. 8, 014001, doi:10. 1088/ 1749-4699/8/1/014001.

Han, J.L.B., Choi, S. (2016), "Motion Effects Synthesis for 4D Films" in *IEEE Transactions on Visualization and Computer Graphics*, Vol. 22, No. 10, pp. 2300-2314.

Kundalakesi, M., Thenmozhi, M. (2018), "Priyadharshini. 7D Holographic Projection Display Technologies", *International Journal of Scientific Research & Growth*, Vol. 6, No. 1, pp. 2321-0613.

Linge, S., Langtangen, H.P. (2020), "Loops and Branching" in *Programming for Computations - Python. Texts in Computational Science and Engineering*, Vol 15, pp. 59-77.

Millman, K.J., Aivazis, M. (2019), "Python for Scientists and Engineers," in *Computing in Science & Engineering*, Vol. 13, No. 2, pp. 9-12.

Oliphant, T.E. (2007), "Python for Scientific Computing" in *Computing in Science & Engineering*, Vol. 9, No. 3, pp. 10-20.

Python$_1$, https://www.fer.unizg.hr/_download/repository/p02-python.pdf. (23.1.2020.).

Python$_2$, https://www.pythonforbeginners.com/random/how-to-use-the-random-module-in-python. (23.1.2020.).

Python$_2$, http://ipaq.petagimnazija.hr/wp-content/uploads/2014/12/Uvod_u_Python.pdf. (20.1.2020.).

Rivers, K., Koedinger, K.R. (2017), "Data-Driven Hint Generation in Vast Solution Spaces: a Self-Improving Python Programming Tutor", *International Journal* of *Artificial Intelligence* in *Education*, Vol. 27, No. 1, pp. 37–64,

Sheng Bao, X.L., Zhang, C. (2011), PyEEG: An Open Source Python Module for EEG/MEG Feature Extraction. 2011. *Hindawi*.

W3SCHOOLS, *Grananje*, https://www.w3schools.com/python/python_ conditions. asp. (21.1.2020).