A mini project report on

SLEEPING BARBER PROBLEM

Second Year of Engineering

In

Computer Engineering

By

Hardik Bhanushali, SE4 - 04

Pratik Bhosale, SE4 - 06

Shivam Chothani, SE4 - 09

Department of Computer Engineering

Shah and Anchor Kutchhi Engineering College, Chembur

University of Mumbai

2018-2019

Table of Content

Chapter 1

Introduction

In computer science, the **sleeping barber problem** is a classic inter-process communication and synchronization problem between multiple operating system processes. The problem is analogous to that of keeping a barber working when there are customers, resting when there are none, and doing so in an orderly manner.

The analogy is based upon a hypothetical barber shop with one barber. The barber has one barber's chair in a cutting room and a waiting room containing a number of chairs in it. When the barber finishes cutting a customer's hair, he dismisses the customer and goes to the waiting room to see if there are others waiting. If there are, he brings one of them back to the chair and cuts their hair. If there are none, he returns to the chair and sleeps in it.

Each customer, when they arrive, looks to see what the barber is doing. If the barber is sleeping, the customer wakes him up and sits in the cutting room chair. If the barber is cutting hair, the customer stays in the waiting room. If there is a free chair in the waiting room, the customer sits in it and waits their turn. If there is no free chair, the customer leaves.

Based on a naïve analysis, the above decisions should ensure that the shop functions correctly, with the barber cutting the hair of anyone who arrives until there are no more customers, and then sleeping until the next customer arrives. In practice, there are a number of problems that can occur that are illustrative of general scheduling problems.

The problems are all related to the fact that the actions by both the barber and the customer (checking the waiting room, entering the shop, taking a waiting room chair, etc.) all take an unknown amount of time. For example, a customer may arrive and observe that the barber is cutting hair, so he goes to the waiting room. While they're on their way, the barber finishes their current haircut and goes to check the waiting room. Since there is no one there (the customer not having arrived yet), he goes back to their chair and sleeps. The barber is now waiting for a customer, but the customer is waiting for the barber. In another example, two customers may arrive at the same time when there happens to be a single seat in the waiting room. They observe that the barber is cutting hair, go to the waiting room, and both attempt to occupy the single chair.

Chapter 2

Problem Definition

Problem : The analogy is based upon a hypothetical barber shop with one barber. There is a barber shop which has one barber, one barber chair, and 4 chairs for waiting for customers if there are any to sit on the chair.

- If there is no customer, then the barber sleeps.
- When a customer arrives, he has to wake up the barber.
- If there are many customers and the barber is cutting a customer's hair, then the remaining customers either wait if there are empty chairs in the waiting room or they leave if no chairs are empty.

Chapter 3

Implementation

#SORCE CODE :

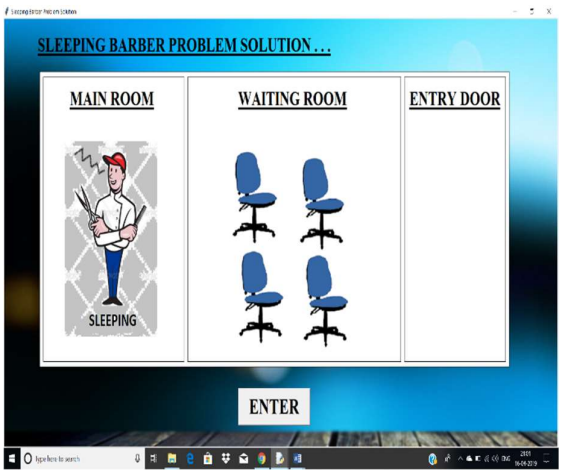
```
from threading import *
import time
from tkinter import *
from PIL import ImageTk , Image

custready=0
access=1
noofseats=4
come=0

def SBP():
    global c
    root=Tk()
    root.title("Sleeping Barber Problem Solution")
    root.geometry("1920x1080")
    c = Canvas(root, bg='cyan', height=1080, width=1920)
    c.pack()
    img = ImageTk.PhotoImage(Image.open("bg.png"))
    c.create_image(0, 0, anchor=NW, image=img)
    c.create_rectangle(100, 100, 1400, 650, fill='white')
    c.create_rectangle(110, 110, 500, 640, fill='white')
    c.create_rectangle(510, 110, 1100, 640, fill='white')
    c.create_rectangle(1110, 110, 1390, 640, fill='white')
    fnt = ('Times', 28, 'bold','underline')
    c.create_text(500, 50, text="SLEEPING BARBER PROBLEM SOLUTION . . .", font=fnt, fill='BLACK')
    fnt = ('Times', 28, 'bold','underline')
    c.create_text(300, 150, text="MAIN ROOM", font=fnt, fill='black')
    c.create_text(800, 150, text="WAITING ROOM", font=fnt, fill='black')
    c.create_text(1250, 150, text="ENTRY DOOR", font=fnt, fill='black')
    fnt = ('Times', 30, 'bold')
    b1 = Button(c, text="ENTER", font=fnt, command= lambda: buttonClick())
    b1.place(x=650, y=690, width=200, height=70)
    root.mainloop()

def buttonClick():
    global come
    come=1

def signal1(s):
    global custready
    custready=s
    custready=custready+1

def wait1(s):
    global custready,c,img19
    cnt=0
    custready=s
    if custready==0:
        print("Barber is Sleeping")
        img1 = ImageTk.PhotoImage(Image.open("sleepingbarber.png"))
```

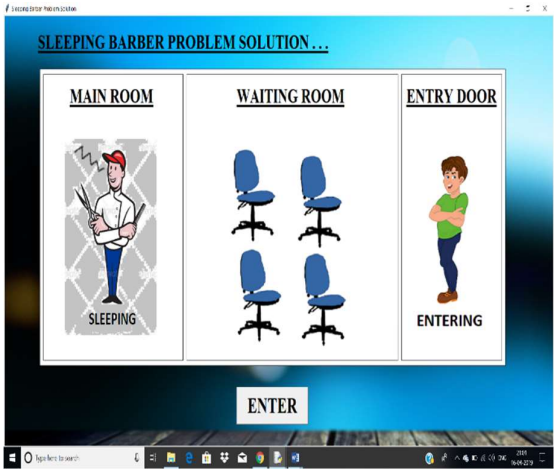
```
        c.create_image(170, 230, anchor=NW, image=img1)
        img2 = ImageTk.PhotoImage(Image.open("chairs.png"))
        c.create_image(620, 250, anchor=NW, image=img2)
    while(custready<=0):
        cnt=1
    if cnt==1:
        cnt=0
        print("Barber Wake Up")
        img3 = ImageTk.PhotoImage(Image.open("wakeupbarber.png"))
        c.create_image(170, 230, anchor=NW, image=img3)
        time.sleep(1)
        img19 = ImageTk.PhotoImage(Image.open("readybarber.png"))
        c.create_image(170, 230, anchor=NW, image=img19)
    custready=custready-1

def signal2(s):
    global access
    access=s
    access=access+1

def wait2(s):
    global access
    access=s
    while(access<=0):
        pass
    access=access-1

class Barber(Thread):
    def run(self):
        global custready,access,noofseats,c
        while(True):
            wait1(custready)
            wait2(access)
            time.sleep(0.2)
            print("Customer Enters into Main Room")
            if custready==1:
                img4 = ImageTk.PhotoImage(Image.open("1person.png"))
                c.create_image(620, 250, anchor=NW, image=img4)
            elif custready==2:
                img5 = ImageTk.PhotoImage(Image.open("2person.png"))
                c.create_image(620, 250, anchor=NW, image=img5)
            elif custready==3:
                img6 = ImageTk.PhotoImage(Image.open("3person.png"))
                c.create_image(620, 250, anchor=NW, image=img6)
            elif custready==4:
                img7 = ImageTk.PhotoImage(Image.open("4person.png"))
                c.create_image(620, 250, anchor=NW, image=img7)
            else:
                img8 = ImageTk.PhotoImage(Image.open("chairs.png"))
                c.create_image(620, 250, anchor=NW, image=img8)
            time.sleep(1)
            noofseats=noofseats+1
            print("started cutting")
            img9 = ImageTk.PhotoImage(Image.open("workingbarber.png"))
            c.create_image(170, 230, anchor=NW, image=img9)
            time.sleep(10)
            print("Cutting complete")
            img10 = ImageTk.PhotoImage(Image.open("readybarber.png"))
            c.create_image(170, 230, anchor=NW, image=img10)
            time.sleep(1)
```

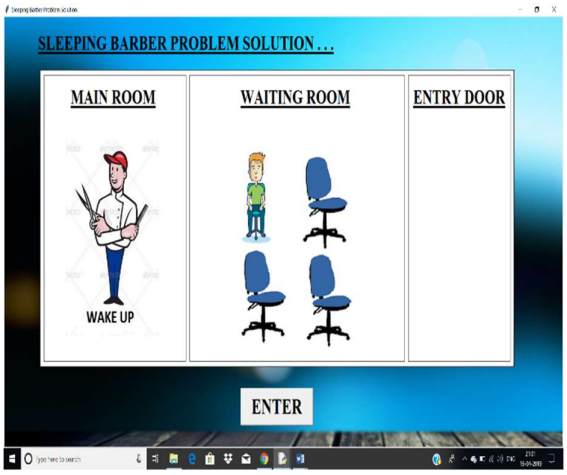
```
            signal2(access)

class Customer(Thread):
    def run(self):
        global custready,access,noofseats,come,c
        while(True):
            if come==1:
                come=0
                img16 = ImageTk.PhotoImage(Image.open("entering.png"))
                i16=c.create_image(1120, 250, anchor=NW, image=img16)
                time.sleep(1)
                c.delete(i16)
                if noofseats>0:
                    print("Customer Enters into Waiting Room")
                    noofseats=noofseats-1
                    signal1(custready)
                    if custready==1:
                        img11 = ImageTk.PhotoImage(Image.open("1person.png"))
                        c.create_image(620, 250, anchor=NW, image=img11)
                    elif custready==2:
                        img12 = ImageTk.PhotoImage(Image.open("2person.png"))
                        c.create_image(620, 250, anchor=NW, image=img12)
                    elif custready==3:
                        img13 = ImageTk.PhotoImage(Image.open("3person.png"))
                        c.create_image(620, 250, anchor=NW, image=img13)
                    elif custready==4:
                        img14 = ImageTk.PhotoImage(Image.open("4person.png"))
                        c.create_image(620, 250, anchor=NW, image=img14)
                    else:
                        img15 = ImageTk.PhotoImage(Image.open("chairs.png"))
                        c.create_image(620, 250, anchor=NW, image=img15)
                    time.sleep(1)
                elif noofseats==0:
                    img17 = ImageTk.PhotoImage(Image.open("nospace.png"))
                    i17=c.create_image(1120, 250, anchor=NW, image=img17)
                    time.sleep(0.5)
                    c.delete(i17)
                    img18 = ImageTk.PhotoImage(Image.open("leaving.png"))
                    i18=c.create_image(1120, 250, anchor=NW, image=img18)
                    time.sleep(1)
                    c.delete(i18)
                    print("Customer Enters But There is No Space, Customer Leaves")

b=Barber()
c=Customer()

b.start()
c.start()

SBP()
```

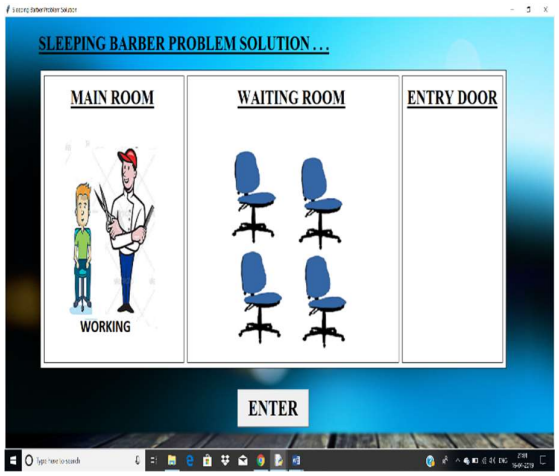
Chapter 4

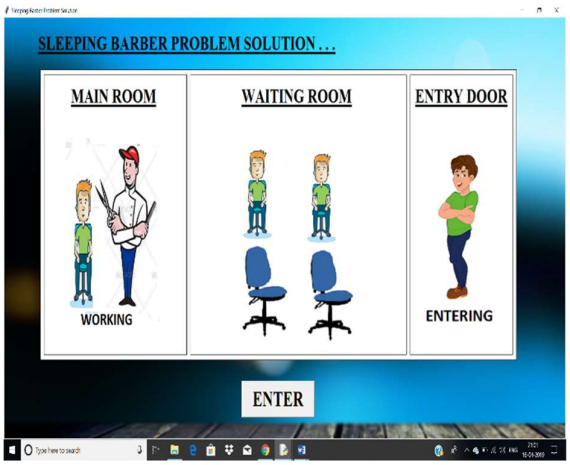
Results

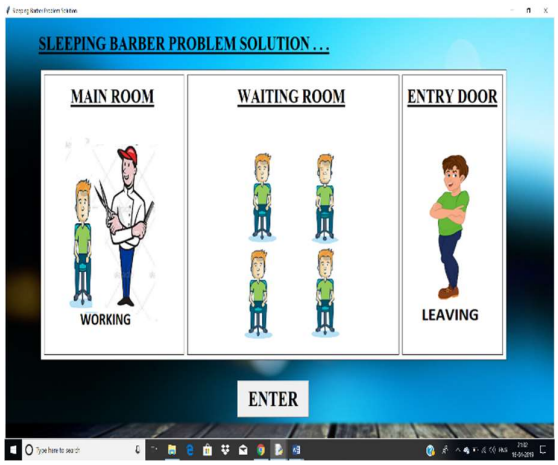












Chapter 5

Conclusion

Thus by using semaphores we successfully solve Sleeping Barber Problem and represent it in the form of GUI. This GUI clearly shows how interconnection and synchronization got placed in between barber and customers which is analogous to multiple processes and resources in operating system.

References

- https://www.geeksforgeeks.org/
- https://en.wikipedia.org/wiki/Sleeping_barber_problem
- youtube