1.

|  |
| --- |
| class Circle(object): |
|  | def\_init\_(self,r): |
|  | self.redius=r |
|  |  |
|  | def area(self): |
|  | return self.darius\*\*2\*3.14 |
|  | aCircle=Circle(2) |
|  | print (aCircle.area()) |

2.

|  |
| --- |
| # Một lớp mô phỏng một hình chữ nhật. |
|  | class Rectangle : |
|  | 'This is Rectangle class' |
|  |  |
|  | # Một phương thức được sử dụng để tạo đối tượng (Contructor). |
|  | def \_\_init\_\_(self, width, height): |
|  |  |
|  | self.width= width |
|  | self.height = height |
|  |  |
|  |  |
|  |  |
|  | def getWidth(self): |
|  | return self.width |
|  |  |
|  |  |
|  | def getHeight(self): |
|  | return self.height |
|  |  |
|  | # Phương thức tính diện tích. |
|  | def getArea(self): |
|  |  |
|  | return self.width \* self.height |

5.

|  |
| --- |
| def reversed\_string(a\_string): |
|  | return a\_string[::-1] |

6.

Top of Form

Bottom of Form

Top of Form

Bottom of Form

|  |  |
| --- | --- |
|  | ss = "Sammy Shark" |
|  | print(ss.upper()) |

7.

|  |
| --- |
| def Tinh(R): |
|  | if R<0: |
|  | print ("Ban kinh khong nho hon 0") |
|  | print ("Ban nhap khong hop le") |
|  | else: |
|  | CV=2\*R\*math.pi |
|  | DT=R\*R\*math.pi |
|  | print ("Chu vi la :",CV) |
|  | print ("Dien tich la :",DT) |
|  |  |
|  | print ("-------Tinh Chu Vi, Dien Tich Hinh Tron---------") |
|  | r=float(input("Nhap ban kinh hinh tron: ")) |
|  | Tinh(r) |

8.ATM

|  |
| --- |
| """ |
|  | Bank renege example |
|  |  |
|  | Covers: |
|  |  |
|  | - Resources: Resource |
|  | - Condition events |
|  |  |
|  | Scenario: |
|  | A counter with a random service time and customers who renege. Based on the |
|  | program bank08.py from TheBank tutorial of SimPy 2. (KGM) |
|  |  |
|  | """ |
|  | import random |
|  |  |
|  | import simpy |
|  |  |
|  |  |
|  | RANDOM\_SEED = 42 |
|  | NEW\_CUSTOMERS = 5 # Total number of customers |
|  | INTERVAL\_CUSTOMERS = 10.0 # Generate new customers roughly every x seconds |
|  | MIN\_PATIENCE = 1 # Min. customer patience |
|  | MAX\_PATIENCE = 3 # Max. customer patience |
|  |  |
|  |  |
|  | def source(env, number, interval, counter): |
|  | """Source generates customers randomly""" |
|  | for i in range(number): |
|  | c = customer(env, 'Customer%02d' % i, counter, time\_in\_bank=12.0) |
|  | env.process(c) |
|  | t = random.expovariate(1.0 / interval) |
|  | yield env.timeout(t) |
|  |  |
|  |  |
|  | def customer(env, name, counter, time\_in\_bank): |
|  | """Customer arrives, is served and leaves.""" |
|  | arrive = env.now |
|  | print('%7.4f %s: Here I am' % (arrive, name)) |
|  |  |
|  | with counter.request() as req: |
|  | patience = random.uniform(MIN\_PATIENCE, MAX\_PATIENCE) |
|  | # Wait for the counter or abort at the end of our tether |
|  | results = yield req | env.timeout(patience) |
|  |  |
|  | wait = env.now - arrive |
|  |  |
|  | if req in results: |
|  | # We got to the counter |
|  | print('%7.4f %s: Waited %6.3f' % (env.now, name, wait)) |
|  |  |
|  | tib = random.expovariate(1.0 / time\_in\_bank) |
|  | yield env.timeout(tib) |
|  | print('%7.4f %s: Finished' % (env.now, name)) |
|  |  |
|  | else: |
|  | # We reneged |
|  | print('%7.4f %s: RENEGED after %6.3f' % (env.now, name, wait)) |
|  |  |
|  |  |
|  | # Setup and start the simulation |
|  | print('Bank renege') |
|  | random.seed(RANDOM\_SEED) |
|  | env = simpy.Environment() |
|  |  |
|  | # Start processes and run |
|  | counter = simpy.Resource(env, capacity=1) |
|  | env.process(source(env, NEW\_CUSTOMERS, INTERVAL\_CUSTOMERS, counter)) |
|  | env.run() |