Creating an engaging mathematics classroom through programming

The following coding examples were prepared by Robin Wang, Preshil Secondary School, for MAV 21

Pattern, sequence, series

Q8 *Find the sum of*

$$\frac{1}{1\times 2}+\frac{1}{2\times 3}+\frac{1}{3\times 4}+\ldots+\frac{1}{99\times 100}$$

Ans: 99/100

```
a=[1/((i+1)*(i+2)) for i in range(99)]
print('The sum is:', sum(a))
```

Q10 *Find the sum of*

$$1\times 1!+2\times 2!+\ldots+9\times 9!$$

Ans: $10!-1$

```
#Q10
def fac(x):
    if x==0:
        return 1
    else:
        return x*fac(x-1)
a=[(i+1)*fac(i+1) for i in range(9)]
print('The sum is:', sum(a))
print('fac(10)-1 is:', fac(10)-1)
```

Two sharks swim towards Mighty Atom at the same time with the same initial speed 1m/s. Shark Bob is 55 metres away from Mighty Atom. His speed increases by 1m/s every second. Shark Alice is 5555 metres away from Mighty Atom. Her speed doubles every 5 metres. Which shark will Mighty Atom meet first?

```
>>> alice=[5/2**i for i in range(50)]
>>> sum(alice)
9.999999999999991
```

Probability

Q10 *A coin is kept tossing until totally three heads appear. What is the expected number of tosses?*

Ans: 6

```
#Q10
import random
n=1000
list_count=[]
for i in range(n):
    list_toss=[]
    count = 0
    while list_toss.count(1)<3:
        a=random.randint(1,2)
        list_toss.append(a)
        count += 1
    list_count.append(count)
ave = sum(list_count)/n
print('The average number of tosses is: {}'.format(ave))
```

Q10 Extension A *A coin is kept tossing until three heads in a row appear. What is the expected number of tosses?*

Ans: 14

```
#Q10 Extension A
import random
n=1000
list_count=[]
for i in range(n):
    list_toss=[random.randint(1,2) for i in range(3)]
    count = 3
    while list_toss[-3::]!=[1,1,1]:
        a=random.randint(1,2)
        list_toss.append(a)
        count += 1
    list_count.append(count)
ave = sum(list_count)/n
print('The average number of tosses is: {}'.format(ave))
```

Q19 *In search of a new car, the player picks a door, say 1. The game host then opens one of the other doors, say 3, to reveal a goat and offers to let player switch from door 1 to door 2. What is the probability that the player picks the new car in the next round by staying with door 1? What is the probability if the player chooses to switch to door 2?*

Ans: 1/3 (non-switching), 2/3 (switching)

```
#Q19
import random
a=[1,1,2] # 2 is the target
n=1000
c1=0
c2=0
for _ in range(n):
    random.shuffle(a)
    aa=a.copy() #non-switching case
    if aa[0]==2:
        c1 += 1
    else:
        c2 += 1 #switching case
print('The probability of non-switching is: {}%, {}/{}'.format(c1/n*100,
    c1, n))
print('The probability of switching is: {}%, {}/{}'.format(c2/n*100, c2,
    n))
```

Q22 *After a nice dinner together, Honda and Skoda play darts to decide who pays the bill. They take turns to throw darts. The first person who hits the Bullseye wins. Honda has a 3/8 winning probability while Skoda's winning probability is 5/8. To compensate for the difference, Honda throws first. Is this a fair game? How many shots do we expect to see until someone wins the game?*

Ans: No (Skoda has a better winning chance of 25/49); 104/49

```
#Q22
import random
def awin():
    if random.random()<3/8:
        return 1
    else:
        return 0
def bwin():
    if random.random()>3/8:
        return 1
    else:
        return 0
n=int(input('Enter the number of runs:'))
win = 0
list_all=[]
j=0 #initialise the number of shots in the 0th game
for i in range(n):
    list_all.append(j)
    j=0 #number of shots
    while True:
        a=awin()
        j+=1
        if a==1:
            win += 1
            break
        b=bwin()
        j+=1
        if b==1:
            break
print('The probability that Honda wins the games is: {}%, {}/{}'.
      format(win/n*100, win, n))
print('The average number of throws per game is: {}'.
      format(sum(list_all)/(n-1)))
```

Relations

Q2 *Ackermann's function is defined for non-negative integers n and k by*

$$\begin{aligned} f(0,n) &= n+1 \\ f(k,0) &= f(k-1,1) \\ f(k+1,n+1) &= f(k,f(k+1,n)) \end{aligned}$$

Evaluate $f(2,2)$.

Ans: 7

```
#Q2
def Ackermann(k, n):
    if k==0:
        return n+1
    if n==0:
        return Ackermann(k-1, 1)
    else:
        return Ackermann(k-1, Ackermann(k, n-1))
print(Ackermann(2,2))
```