Q.1 What possible output from the given options is expected to be displayed when the following Python code is executed? (Marks: 1) import random Signal=['RED', 'YELLOW', GREEN'] for K in range (2, 0, -1) : R = random.randrange (K) print (Signal[R], end = '#') (a) YELLOW#RED# (c) GREEN#RED# (b) RED#GREEN# (d) YELLOW#GREEN#

In [4]:
```
import random
Signal=['RED', 'YELLOW', 'GREEN']
for K in range (2, 0, -1) :
        R = random.randrange (K)
        print (Signal [R], end = '#')
```

```
YELLOW#RED#
```

In [ ]:

Q.2 Observe the given Python code carefully: a=20 def convert(a): b=20 a=a+b convert(10) print(a) Select the correct output from the given options: (Marks: 1) (a) 10 (c) 30 (b) 20 (d) Error

In [5]:
```
a=20
def convert(a):
        b=20
        a=a+b
convert(10)
print(a)
```

```
20
```

In [ ]:

Q.3 The code given below accepts five numbers and displays whether they are even or odd: Observe the following code carefully and rewrite it after removing all syntax and logical errors: Underline all the corrections made. (Marks: 2) def Evenodd() for i in range (5): num=int(input("Enter a number") if num/2==0: print("Even") else: print("Odd") EvenOdd()

In [ ]:
```
def EvenOdd():
        for i in range (5):
                num=int(input("Enter a number"))
                if num%2==0:
                        print("Even")
                else:
                        print("Odd")
EvenOdd()
```

In [ ]:

Q.4 Write a user defined function in Python named showGrades(S) which takes the dictionary S as an argument. The dictionary, S contains Name: [Eng, Math, Science] as key: value pairs. The function displays the corresponding grade obtained by the students according to the following grading rules: Average of Eng, Math, Science Grade >=90 A <90 but >=60 B <60 C For example : Consider the following dictionary S={"AMIT": [92,86,64], "NAGMA": [65,42,43],"DAVID": [92,90,88]} The output should be: AMIT - B NAGMA-C DAVID - A (Marks:2)

In [8]:
```
def calculate_grade(score):
    if score >= 90:
        return 'A'
    elif score >= 60:
        return 'B'
```

```
    else:
        return 'C'

def showGrades(S):
    for name, scores in S.items():
        average_score = sum(scores) / len(scores)
        grade = calculate_grade(average_score)
        print(f"{name} - {grade}", end="\t")


S = {"AMIT": [92, 86, 64], "NAGMA": [65, 42, 43], "DAVID": [92, 90, 88]}
showGrades(S)
```

```
AMIT - B        NAGMA - C       DAVID - A
```

In [ ]:

Q.5 Write a user defined function in Python named Puzzle(W, N) which takes the argument W as an English word and N as an integer and returns the string where every Nth alphabet of the word w is replaced with an underscore ("_"). For example: if W contains the word "TELEVISION" and N is 3, then the function should return the string "TE_EV_SI_N". Likewise for the word "TELEVISION" if N is 4, t hen the function should return "TEL_VIS_ON". (Marks: 2)

In [18]:
```
def Puzzle(W, N):
    new_word = ""
    for i in range(len(W)):
        if (i + 1) % N == 0:
            new_word += "_"
        else:
            new_word += W[i]
    return new_word


word = "TELEVISION"
N = 3
result = Puzzle(word, N)
print(result)

N = 4
result = Puzzle(word, N)
print(result)
```

```
TE_EV_SI_N
TEL_VIS_ON
```

In [ ]:

Q.6 Write the output displayed on execution of the following Python code: (Marks: 2) LS=["HIMALAYA", "NILGIRI", "ALASKA", "ALPS"] D={} for S in LS : if len(S)%4 = 0: D[S] = len(S) for K in D: print(K,D[K],sep="#")

In [15]:
```
LS=["HIMALAYA", "NILGIRI", "ALASKA", "ALPS"]
D={}
for S in LS :
        if len(S)%4 == 0:
                D[S] = len(S)
for K in D:
        print(K,D[K],sep="#")
```

```
HIMALAYA#8
ALPS#4
```

In [ ]:

Q.7 Predict the output of the following code: (Marks: 2) def callon (b=20, a=10): b=b+a a=b-a print (b, "#",a) return b x=100 y=200 x=callon (x,y) print(x, "@",y) y=callon (y) print (x, "@",y)

In [11]:
```
def callon (b=20, a=10):
        b=b+a
        a=b-a
        print (b, "#",a)
        return b
x=100
y=200

x=callon(x,y)
print(x, "@",y)
y=callon(y)
print (x, "@",y)
```

```
300 # 100
300 @ 200
210 # 200
300 @ 210
```

In [ ]:

Q.8 Write the output on execution of the following Python code: (Marks: 3) S="Racecar Car Radar" L=S.split() for W in L: x=W. upper() if x=x[::-1]: for I in x: print (I, end="*") else: for I in W: print (I, end="#") print()

In [16]:
```
S="Racecar Car Radar"
L=S.split()

for W in L:
        x=W.upper()
        if x==x[::-1]:
                for I in x:
                        print(I, end="*")
        else:
                for I in W:
                        print (I, end="#")
        print()
```

```
R*A*C*E*C*A*R*
C#a#r#
R*A*D*A*R*
```

In [ ]:

Q.9 Consider a list named Nums which contains random integers. Write the following user defined functions in Python and perform the specified operations on a stack named BigNums. (Marks: 3) (i) PushBig(): It checks every number from the list Nums and pushes all such numbers which have 5 or more digits into the stack, BigNums. (ii) PopBig(): It pops the numbers from the stack, BigNums and displays them. The function should also display "Stack Empty" when there are no more numbers left in the stack. For example: If the list Nums contains the following data: Nums=[213,10025,167,254923,14,1297653,31498,386,92765] Then on execution of PushBig(), the stack BigNums should store : [10025, 254923, 1297653, 31498, 92765] And on execution of PopBig (), the following output should be displayed: 92765 31498 1297653 254923 10025 Stack Empty

In [14]:
```python
def push(stack, item):
    stack.append(item)

def pop(stack):
    if not is_empty(stack):
        return stack.pop()

def is_empty(stack):
    return len(stack) == 0

def PushBig(Nums, BigNums):
    for num in Nums:
        if len(str(num)) >= 5:
            push(BigNums, num)

def PopBig(BigNums):
    while not is_empty(BigNums):
        print(pop(BigNums))
    print("Stack Empty")

# Example usage:
Nums = [213, 10025, 167, 254923, 14, 1297653, 31498, 386, 92765]
BigNums = []

# Push numbers with 5 or more digits into BigNums
PushBig(Nums, BigNums)

# Pop and display numbers from BigNums
PopBig(BigNums)
```

```
92765
31498
1297653
254923
10025
Stack Empty
```

In [ ]: