In [2]:

```python
# Program 1
# Constructing output list Without Using List comprehensions
input_list = [1,2,3,4,5,6,7,8]
output_list = []
# using Loop for constructing output list
for var in input_list:
    if var % 2==0:
        output_list.append(var)
print("output list  using for loop:",output_list)
```

```
output list  using for loop: [2, 4, 6, 8]
```

In [3]:

```python
# Program 2
# Using list comprehensions for constructing output list
input_list = [1,2,3,4,5,6,7,8]
list_using_comp = [var for var in input_list if var % 2==0]
print("output list using list comprehensions:",list_using_comp)
```

```
output list using list comprehensions: [2, 4, 6, 8]
```

In [5]:

```python
# Program 3
#Constructing ouput list using for loop
output_list = []
for var in range(1,10):
    output_list.append(var ** 2)
print("Output List using for loop:", output_list)

```

```
Output List using for loop: [1, 4, 9, 16, 25, 36, 49, 64, 81]
```

In [8]:

```python
# Program 4
# Constructing output list using list comprehension
list_using_comp = [var **2  for var in range(1,10)]
print("Output List Using list comprehension:",list_using_comp)
```

```
Output List Using list comprehension: [1, 4, 9, 16, 25, 36, 49, 64, 81]
```

In [10]:

```python
# Program 5
num_list = [ y for y in range(100) if y % 2==0 if y % 5==0]
print(num_list)
```

```
[0, 10, 20, 30, 40, 50, 60, 70, 80, 90]
```

In [14]:

```python
# Program 6
transposed = []
matrix= [[1,2,3,4],[4,5,6,8]]
for i in range(len(matrix[0])):
    transposed_row= []
    for row in matrix:
        transposed_row.append(row[i])
    transposed.append(transposed_row)
print(transposed)
```

```
[[1, 4], [2, 5], [3, 6], [4, 8]]
```

In [18]:

```
# Program 7
matrix = [[1,2],[3,4],[5,6],[7,8]]
transpose= [[row[i] for row in matrix ] for i in range(2)]
print(transpose)
```

```
[[1, 3, 5, 7], [2, 4, 6, 8]]
```

In [22]:

```
# Program 8
dict1 ={'a':1,'b':2,'c':3,'d':4,'e':5}
#Double each value in dictionary
double_dict1 = {k:v*2 for (k,v) in dict1.items()}
print(double_dict1)

```

```
{'a': 2, 'b': 4, 'c': 6, 'd': 8, 'e': 10}
```

In [26]:

```
# Program 9
dict1 = {'a': 1,'b':2,'c':3,'d':4,'e':5}
#Double each value in the dictionary
double_dict1 = {k*2:v*3 for (k,v) in dict1.items()}
print(double_dict1)
```

```
{'aa': 3, 'bb': 6, 'cc': 9, 'dd': 12, 'ee': 15}
```

In [28]:

```
# Program 10
dict1 = {'a':1, 'b':2, 'c':3, 'd':4, 'e':5, 'f':6, 'g':7}
dict1_doubleCond = {k:v for (k,v) in dict1.items() if v>2 if v%2==0}
print(dict1_doubleCond)
```

```
{'d': 4, 'f': 6}
```

In [29]:

```
# Program 11
nested_dict = {'first':{'a':1}, 'second ':{'b':2}, 'third':{'c':3},'forth':{
float_dict = {outer_k: {float(inner_v) for (inner_k, inner_v) in outer_v.ite
print(float_dict)
```

```
{'first': {1.0}, 'second ': {2.0}, 'third': {3.0}, 'forth': {4.0}}
```

In [30]:

```
# Program 12
# python code to illustrate cube of a number showing difference between def(
def cube(y):
    return y*y*y;
c= lambda x: x*x*x
print(c(7))
print(cube(5))
```

```
343
125
```

In [31]:

```
# Program 13
# Python code to illustrate filter() with lambda()
li = [5,7,22,97,54,62,77,23,73,61]
final_list = list(filter(lambda x:(x%2 !=0),li))
print(final_list)
```

```
[5, 7, 97, 77, 23, 73, 61]
```

In [36]:

```
# Program 14
# Python code to illustrate reduce() with lambda() to get double of a list
li = [5,7,22,97,54,62,77,23,73,61]
final_list = list(map(lambda x: x*2, li))
print(final_list)
```

```
[10, 14, 44, 194, 108, 124, 154, 46, 146, 122]
```

In [35]:

```
# Program 15
# Python code to illustrate reduce() with lambda() to get sum of a list
from functools import reduce
li=[5,8,10,20,50,100]
sum = reduce((lambda x, y:x+y),li)
print(sum)
```

```
193
```