Assignment

1. Functions

a. Create a function that computes the factorial of a given number. Similar to what we saw before sum(7,8)=15, your function should output the factorial for a given number, say factorial(3)=6. Factorial is nothing but 1x2x3

b. Let's make an extension to the above created function. Create a new function that prints all the factors of a number less than the given number. Eg: if your function name is factorial. Then factorial 2(4) should print – 24, 6, 2, 1

In [14]:
```
def factorial(num):
    '''
    A function that computes the factorial of a given number.
    '''
    fact = 1
    try:
        for i in range(num,1,-1):
            fact *= i
        return fact
    except Exception as e:
        print(f'Error : {e}')

factorial(3)
```

Out[14]: 6

In [18]:
```
def factorial2(num):
    '''
    A function that prints all the factors of a number less than the given number.
    '''
    print_fact = []
    try:
        for i in range(num,0,-1):
            print_fact.append(factorial(i))
        return print_fact
    except Exception as e:
        print(f'Error : {e}')

factorial2(4)
```

Out[18]:
```
[24, 6, 2, 1]
```

2. Let's play a game – Guess by Birthday

The goal of the game is to guess my birthday (between 1-31) and the output from your program should be the number of trials or guesses your program made before you got it right.

Eg:

Input – my birthdate, a number from 1 to 31

Output – No. of trials you made before you guessed it right

Say my function name is guess_me i.e guess_me(birthdate) will be my call to the function. My algorithm can be like this –

Start with 1, compare it with the given birthday and then check with 2, then with 3, then 4, 5…. Until I match the given date. Then output the number of loops I did.

So when I run guess_me(22) my above algorithm will output 22.

The above algorithm is a brute force one, it's not optimized to get the current answers efficiently. You can come up with better algos. The answer will keep varying and it's an open ended question to figure out your own algo to minimize the number of guesses you might need.

In [48]:

```python
import random as rn

def guess_me(birthdate):
    '''
    To guess my birthday (between 1-31) and
    the output from your program should be
    the number of trials or guesses your program
    made before you got it right.
    '''
    if 1 <= birthdate <= 31:
        guess_birthdate, counter = rn.randint(1, 31), 0

        while 1:
            if (guess_birthdate == birthdate):
                print(f'Your birthdate is {guess_birthdate} \nNumber of guess is : {counter+1}')
                break
            else:
                if (guess_birthdate > birthdate):
                    print(f'Number of guess is : {counter+1} || Guess is {guess_birthdate} .')
                    guess_birthdate = rn.randint(1, guess_birthdate)
                    counter += 1
                else:
                    print(f'Number of guess is : {counter+1} || Guess is {guess_birthdate} .')
                    guess_birthdate = rn.randint(guess_birthdate, 31)
                    counter += 1
    else:
        print('Enter valid birthdate (1 to 31) .')


guess_me(22)
```

```
Number of guess is : 1 || Guess is 29 .
Number of guess is : 2 || Guess is 25 .
Number of guess is : 3 || Guess is 10 .
Number of guess is : 4 || Guess is 15 .
Number of guess is : 5 || Guess is 20 .
Your birthdate is 22
Number of guess is : 6
```

3. Data scraping

Scrape below website player data and push it into the excel sheet. it should contains all the columns starting from [Mat to 6s]

https://www.iplt20.com/stats/2021/most-runs (https://www.iplt20.com/stats/2021/most-runs)

In [65]:

```
import pandas as pd
import requests
from bs4 import BeautifulSoup as bs
from urllib.request import urlopen as uReq
from selenium import webdriver
import time

import warnings
warnings.filterwarnings("ignore")

```

In [68]:

```
url = "https://www.iplt20.com/stats/2021/most-runs"

# initiating the webdriver.
driver = webdriver.Chrome('E:\Basic Software Setup\chromedriver.exe')
driver.get(url)

# load all data by clicking 'View All' button
btn_click = driver.find_element_by_css_selector(".np-mostrunsTab__btn a")

#perform click with execute_script
driver.execute_script("arguments[0].click();", btn_click);
time.sleep(1)

table1 = driver.find_element_by_xpath('/html/body/div[2]/section/div/div/div[2]/div[3]/div[2]/div[1]/table')
all_data = table1.text

# collecting the data
row_len = len(all_data.split('\n'))
all_li = [all_data.split('\n')[i].split(' ') for i in range(3,row_len,3)]
player_li = [all_data.split('\n')[i] for i in range(2,row_len,3)]
table_cols = all_data.split('\n')[0].split(' ')

# inseting data in dataframe
df_ipl = pd.DataFrame(all_li, columns = table_cols[2:])
df_ipl['Player'] = player_li

# shift column 'Player' to first position
first_column = df_ipl.pop('Player')
df_ipl.insert(0, 'Player', first_column)

df_ipl.to_csv('ipl_most_runs.csv', index = False)

driver.close()
display(df_ipl.head())
```

	Player	Mat	Inns	NO	Runs	HS	Avg	BF	SR	100	50	4s	6s
0	Jos Buttler	14	14	1	629	116	48.38	428	146.96	3	3	56	37
1	K L Rahul	14	14	3	537	103*	48.82	397	135.26	2	3	42	25
2	Quinton De Kock	14	14	1	502	140*	38.62	336	149.40	1	3	47	22
3	Shikhar Dhawan	14	14	2	460	88*	38.33	375	122.66	0	3	47	12

	Player	Mat	Inns	NO	Runs	HS	Avg	BF	SR	100	50	4s	6s
4	Faf Du Plessis	14	14	1	443	96	34.08	339	130.67	0	3	46	13

In [72]:
```
df_ipl.iloc[:,1:].to_csv('ipl_most_runs_1.csv', index = False)
df_ipl.iloc[:,1:].head()
```

Out[72]:

	Mat	Inns	NO	Runs	HS	Avg	BF	SR	100	50	4s	6s
0	14	14	1	629	116	48.38	428	146.96	3	3	56	37
1	14	14	3	537	103*	48.82	397	135.26	2	3	42	25
2	14	14	1	502	140*	38.62	336	149.40	1	3	47	22
3	14	14	2	460	88*	38.33	375	122.66	0	3	47	12
4	14	14	1	443	96	34.08	339	130.67	0	3	46	13

In [ ]:
```

```