In [ ]:
```
##Assignment 3
```

In [ ]:
```
#Question1
#1. Write a class called RestaurantCheck. It should have the following: (use OOPs concepts)

 # • Fields called check_number, sales_tax_percent, subtotal, table_number, and server_name representing an identification for the       check, the bill without tax added, the sa
 # • A constructor that sets the values of all four fields
 #  • A method called calculate_total that takes no arguments (besides self) and returns the total bill including sales tax.
 #  • A method called print_check that writes to a file called check###.txt, where ### is the check number and writes information about      the check to that file, formatted like

 #    Check Number: 443
 #    Sales tax: 6.0%
 #    Subtotal: $23.14
 #    Total: $24.53
  #   Table Number: 17
 #     Server: Sonic the Hedgehog
 #
 #  Test the class by creating a RestaurantCheck object and calling the print_check() method.
```

In [44]:
```
class RestaurantCheck:
    total_bill=0
    filename=""
    def __init__(self,check_number,sales_tax_percent,subtotal,table_number,server_name):
        self.check_number=check_number
        self.sales_tax_percent=sales_tax_percent
        self.subtotal=subtotal
        self.table_number=table_number
        self.server_name=server_name

    def calculate_total(self):
        total_bill=self.subtotal+(self.subtotal*(self.sales_tax_percent/100))
        return round(total_bill,2)

    def print_check(self):
        filename="check"+str(self.check_number)+'.txt'
        with open(filename,'w') as f:
            f.write('Check Number: ' +str(self.check_number) +'\n')
            f.write('Sales tax: ' +str(self.sales_tax_percent) +'\n')
            f.write('Subtotal: $' +str(self.subtotal) +'\n')
            f.write('Total: $' +str(self.calculate_total()) +'\n')
            f.write('Table Number: '+str(self.table_number) +'\n')
            f.write('Server: '+str(self.server_name) +'\n')
```

In [47]:
```
Res1=RestaurantCheck(443,6,23.14,17,"Sonic the Hedgehog")
#print(Res1.calculate_total())
Res1.print_check()
```

In [48]:
```
Res2=RestaurantCheck(444,6,25.14,18,"Sonic")
#print(Res1.calculate_total())
Res2.print_check()
```

In [ ]:
```
#Question2
#Write a Regular Expression Python function to Validate
     #Phone No, (Must be 10 digits)
     #Name, (first Char must be uppercase)
     #E-Mail, (abc@abc.com)
     #Date .(DD-MM-YYYY)
```

In [49]:
```
import re
def validate_phone_number(phone_number):
    if re.match(r'^\d{10}$', phone_number):
        return True
    else:
        return False
def validate_name(name):
    if re.match(r'^[A-Z][a-zA-Z\s]*$', name):
        return True
    else:
        return False

def validate_email(email):
    if re.match(r'^[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$', email):
        return True
    else:
        return False

def validate_date(date):
    if re.match(r'^\d{2}-\d{2}-\d{4}$', date):
        return True
    else:
        return False
```

In [50]:
```
phone_number = "9908405234"
name = "Yogesh War"
email = "yogeshbugide@gmail.com"
date = "05-03-1994"
```

In [52]:
```
if validate_phone_number(phone_number):
    print("Phone Number is containing 10 digits")
else:
    print("Phone NUmber is not containing 10 digits")
```

```
Phone Number is containing 10 digits
```

In [54]:
```
phone_number = "990840523"
```

In [55]:
```
if validate_phone_number(phone_number):
    print("Phone Number is containing 10 digits")
else:
    print("Phone NUmber is not containing 10 digits")
```

```
Phone NUmber is not containing 10 digits
```

In [56]:
```
if validate_name(name):
    print("First Letter of name is Upper case")
else:
    print("First Letter of name is not Upper case")
```

```
First Letter of name is Upper case
```

In [57]:
```
name = "yogesh War"
```

In [58]:
```
if validate_name(name):
    print("First Letter of name is Upper case")
else:
    print("First Letter of name is not Upper case")
```

```
First Letter of name is not Upper case
```

In [59]:
```
if validate_email(email):
    print("It is a valid Email ID")
else:
    print("It is not a valid Email ID")
```

```
It is a valid Email ID
```

In [60]:
```
email = "yogeshbugide@@gmail.com"
```

In [61]:
```
if validate_email(email):
    print("It is a valid Email ID")
else:
    print("It is not a valid Email ID")
```

```
It is not a valid Email ID
```

In [62]:
```
if validate_date(date):
    print("It is a valid Date")
else:
    print("It is not a valid Date")
```

```
It is a valid Date
```

In [65]:
```
date = "3-03-1994"
```

In [66]:
```
if validate_date(date):
    print("It is a valid Date")
else:
    print("It is not a valid Date")
```

```
It is not a valid Date
```

In [ ]:
```

```