String Format dan File

Perkuliahan Algoritma Pemrograman pada Semester Ganjil 2018

String Format

Display data juga memegang peranan penting dalam pembuatan *code*, karena *display* yang tepat akan memudahkan *user* untuk lebih memahami *output* yang dihasilkan oleh suatu *code*. Untuk menghasilkan display yang tepat ini, terkadang programmer menyatukan antara tipe data string dengan tipe data yang lain, serta mengatur tampilan string tersebut.

Untuk menampilkan antara data dengan tipe data string dengan suatu data yang memiliki tipe data numerik, dapat dilakukan dengan cara sebagai berikut:

In [ ]:
```
num=100
print('Jumlah mahasiswa dalam kelas ini adalah ',num,' mahasiswa')
```

Cara lain yang dapat ditempuh oleh programmer untuk mengatur tampilan data dapat dilakukan dengan cara *string format*. Dengan *formatting string*, maka programmer dapat mengatur tampilan string, menggabungkan string dengan tipe data yang lain. Untuk menunjukkan tipe data ini dapat dilakukan dengan menggunak *string format*, sebagai berikut :

- %d
- %s
- %f

Contoh penggunaan *formatting string* ini adalah :

In [ ]:
```
num = 40
text="jumlah mahasiswa dalam kelas ini sama dengan %d mahasiswa" % num
print(text)
```

In [ ]:
```
kelas="AP"
text="Jumlah mahasiswa dalam kelas %s adalah %d mahasiswa" %(kelas,num)
print(text)
```

Programmer dapat juga mengatur lebar data yang akan ditampilkan dengan mengatur spasi pada *string format* tersebut. Contoh pengaturan spasi dapat dilakukan sebagai berikut :

In [2]:
```
num = 4
text="jumlah mahasiswa dalam kelas ini adalah %d mahasiswa" % num
print(text)
text="jumlah mahasiswa dalam kelas ini adalah %2d mahasiswa" % num
print(text)
text="jumlah mahasiswa dalam kelas ini adalah %3d mahasiswa" % num
print(text)
text="jumlah mahasiswa dalam kelas ini adalah %5d mahasiswa" % num
print(text)
```

```
jumlah mahasiswa dalam kelas ini adalah 4 mahasiswa
jumlah  mahasiswa  dalam  kelas ini  adalah   4 mahasiswa
jumlah  mahasiswa  dalam  kelas ini  adalah    4  mahasiswa
jumlah  mahasiswa  dalam  kelas ini  adalah      4  mahasiswa
```

In [3]:
```
kelas="Matematika"
text="Jumlah mahasiswa dalam kelas %s adalah %d mahasiswa" %(kelas,num)
print(text)
text="Jumlah mahasiswa dalam kelas %2s adalah %d mahasiswa" %(kelas,num)
print(text)
text="Jumlah mahasiswa dalam kelas %5s adalah %d mahasiswa" %(kelas,num)
print(text)
```

```
Jumlah mahasiswa dalam kelas Matematika adalah 4 mahasiswa
Jumlah mahasiswa dalam kelas Matematika adalah 4 mahasiswa
Jumlah mahasiswa dalam kelas Matematika adalah 4 mahasiswa
```

Untuk data yang bertipe data float, dapat ditampilkan sebagai integer ataupun sebagai bilangan pecahan (real), seperti contoh berikut

In [ ]:
```
num=32/5
text="bilangan pecahan sebagai integer=%d" %num
print(text)
```

In [7]:
```
num=22/7
text="bilangan pecahan=%f%s%s" %(num,' ','test')
print(text)
text="bilangan pecahan=%f%5s%s" %(num,' ','test')
print(text)
```

```
bilangan pecahan=3.142857 test
bilangan pecahan=3.142857      test
```

Programmer dapat juga mengatur jumlah angka dibelakang koma, dengan cara sebagai berikut

In [5]:
```
num=22/7
text="bilangan pecahan=%f" %num
print(text)
text="bilangan pecahan=%2.3f" %num
print(text)
text="bilangan pecahan=%4.3f" %num
print(text)
text="bilangan pecahan=%6.3f" %num
print(text)
text="bilangan pecahan=%10.2f" %num
print(text)
text="bilangan pecahan=%10.3f" %num
print(text)
```

```
bilangan pecahan=3.142857
bilangan pecahan=3.143
bilangan pecahan=3.143
bilangan pecahan= 3.143
bilangan  pecahan=       3.14
bilangan  pecahan=      3.143
```

File

Agar data tetap tersimpan dan dapat diupdate sewaktu-waktu, maka data tersebut dapat disimpan dalam suatu file diluar file *code*. Untuk menggunakan file ini, terdapat tiga operasi penting, yaitu :

- open, setiap file yang akan digunakan dalam file code, harus dipanggil terlebih dahulu dengan menggunakan perintah open. Tanpa perintah ini, maka file tidak dapat digunakan untuk operasi berikutnya
- write, setelah data diproses, data dapat disimpan kembali ke dalam file dengan menggunakan perintah write
- close, tutuplah file dengan menggunakan perintah ini.

Berikut adalah contoh penggunakan data di dalam file

In [3]:
```
data = open("Hello1.txt", "a") # a for append ; w for write
temp="Hello World"
data.write("%s\n"%temp)
data.close()
```

Coba penggunaan parameter *append* (a) dan *write* (w), dan jelaskan perbedaan kedua parameter tersebut pada file.

Berikut adalah contoh menyimpan data dalam suatu file dan mengambil data yang terdapat pada suatu file

In [ ]:
```
def createFile():
    file=open("students.txt","w")
    for i in range(0,4):
        nim=input("nim=")
        nama=input("nama=")
        file.write("%s %s \n " % (nim,nama))
    file.close()



createFile()
```

In [ ]:
```
def readFile():
    file=open("students.txt")
    data={}
    for eachLine in file:
        temp=eachLine
        (nim,nama)=temp.split()
        data[nim]=nama

    return data
```

In [ ]:
```
mhs=readFile()
print(mhs)
```

In [ ]:
```
def addNilai(data):
    nilai={}
    for i in data.keys():
        temp=int(input('Masukkan nilai dari %s \n'% data[i]))
        nilai[i]=[data[i],temp]
    return nilai
```

In [ ]:
```
nilai=addNilai(mhs)
```

In [ ]:
```
print(nilai)
```

In [ ]:

```
def saveFile(data):
    file=open("nilai.txt","w")
    for i  in data.keys():
        text="%s  %10s  %10d  \n" % (i,data[i][0],data[i][1])
        file.write(text)
    file.close()

saveFile(nilai)
```

Latihan

Buat fungsi untuk menghitung nilai rata-rata mahasiswa, dan nilai maksimum yang diperoleh mahasiswa

In [ ]:

```
a={}
a['001']=['rita',90]
a['002']=['doni',78]
print(a)
b={}
b['001']='dian'
b['002']='reni'
print(b['002'])
```