Untitled5

October 19, 2020

[11]:
```
#DEFINISI TYPE
# type pecahan : <n:integer, d:integer>0>
# <n,d> adalah sebuah pecahan dengan n sebagai pembilang (numerator) dan d␣
 ↪sebagai penyebut (denumerator)
#DEFINISI DAN SPESIFIKASI KONSTRUKTOR
# MakeP : integer, integer > 0 --> pecahan
# MakeP(n,d) membuat pecahan dari pembilang n dan penyebut d
class pecahan:
    def __init__(self,n,d):
        self.n = n
        self.d = d

#DEFINISI DAN SPESIFIKASI SELEKTOR
# Pemb : pecahan --> integer
# Pemb(P) mengembalikan pembilang n dari pecahan P
#REALISASI dalam Python
def Pemb(P):
    return P.n

# Peny : pecahan --> integer > 0
# Peny(P) mengembalikan penyebut d dari pecahan P
#REALISASI dalam Python
def Peny(P):
    return P.d

#DEFINISI DAN SPESIFIKASI OPERATOR TERHADAP PECAHAN
# AddP : 2 pecahan --> pecahan
# AddP(P1,P2)menjumlahkan pecahan P1 dan P2, hasilnya adalah pecahan
#REALISASI dalam Python
def AddP(P1,P2):
    return pecahan((Pemb(P1)*Peny(P2)) + (Pemb(P2)*Peny(P1)) ,␣
 ↪Peny(P1)*Peny(P2))

# SubP : 2 pecahan --> pecahan
# SubP(P1,P2) mengurangkan pecahan P1 dengan P2, hasilnya adalah pecahan
#REALISASI dalam Python
def SubP(P1,P2):
```

```
    return pecahan((Pemb(P1)*Peny(P2)) - (Pemb(P2)*Peny(P1)) ,␣
↪Peny(P1)*Peny(P2))

# MulP : 2 pecahan --> pecahan
# MulP(P1,P2) mengalikan pecahan P1 dengan P2, hasilnya adalah pecahan
#REALISASI dalam Python
def MulP(P1,P2):
    return pecahan((Pemb(P1)*Pemb(P2)) , (Peny(P1)*Peny(P2)))

# DivP : 2 pecahan --> pecahan
# DivP(P1,P2) membagi pecahan P1 dengan P2, hasilnya adalah pecahan
#REALISASI dalam Python
def DivP(P1,P2):
    return pecahan((Pemb(P1)*Peny(P2)) , (Pemb(P2)*Peny(P1)))

# RealP : pecahan --> real
# Real(P) mengubah pecahan P ke dalam nilai real
#REALISASI dalam Python
def RealP(P):
    return Pemb(P) / Peny(P)

# CetakP : pecahan --> string
# CetakP(P) mencetak nilai elemen pecahan P ke layar
#REALISASI dalam Python
def CetakP(P):
    print("Pembilang: ", Pemb(P), "\nPenyebut: ", Peny(P))

#aplikasi
CetakP(AddP(pecahan(3,4),pecahan(1,2)))
CetakP(SubP(pecahan(3,4),pecahan(1,2)))
CetakP(MulP(pecahan(3,4),pecahan(1,2)))
CetakP(DivP(pecahan(3,4),pecahan(1,2)))
print(RealP(pecahan(3,4)))
```

```
Pembilang:  10
Penyebut:  8
Pembilang:  2
Penyebut:  8
Pembilang:  3
Penyebut:  8
Pembilang:  6
Penyebut:  4
0.75
```

[ ]:

[12]:
```
#DEFINISI TYPE
# type pecahanc : <bil: integer, n:integer >=0, d:integer >0>
# <bil,n,d> adalah sebuah pecahan campuran dengan bil sebagai bilangan bulat,
  # n sebagai pembilang (numerator) dan d sebagai penyebut (denumerator) dimana␣
 ↪n < d
#DEFINISI DAN SPESIFIKASI KONSTRUKTOR
# MakePC : integer, integer >= 0, integer > 0 --> pecahan
# MakeP(bil,n,d) membuat pecahan campuran dari bilangan bulat bil, pembilang n␣
 ↪dan penyebut d
class pecahanc :
    def __init__(self,bil,n,d):
        self.bil = bil
        self.n = n
        self.d = d

#DEFINISI DAN SPESIFIKASI SELEKTOR
# Bil : pecahanc --> integer
# Bil(P) mengembalikan bilangan bulat bil dari pecahan campuran P
#REALISASI dalam Python
def Bil(P):
    return P.bil

# Pembc : pecahanc --> integer >= 0
# Pembc(P) mengembalikan pembilang n dari pecahan campuran P
#REALISASI dalam Python
def Pembc(P):
    return P.n

# Penyc : pecahanc --> integer > 0
# Penyc(P) mengembalikan penyebut d dari pecahan campuran P
#REALISASI dalam Python
def Penyc(P):
    return P.d


#DEFINISI DAN SPESIFIKASI OPERATOR TERHADAP PECAHAN Campuran
# KonversiPCtoP : pecahanc --> pecahan
# KonversiPCtoP(P) mengkonversi pecahan campuran P menjadi pecahan biasa
#REALISASI dalam Python
def KonversiPCtoP(P):
    if Bil(P) >= 0:
        return pecahan((Bil(P)*Penyc(P) + Pembc(P)), Penyc(P))
    else:
        return pecahan((Bil(P)*Penyc(P) - Pembc(P)), Penyc(P))

# KonversiPtoPC : pecahan --> pecahanc
# KonversiPtoPC(P) mengkonversi pecahan biasa P menjadi pecahan campuran
```

```python
#REALISASI dalam Python
def KonversiPtoPC(P):
    if Pemb(P) >= 0:
        return pecahanc(Pemb(P) // Peny(P), Pemb(P) % Peny(P), Peny(P))
    else:
        return pecahanc(-1 * (abs(Pemb(P)) // Peny(P)), abs(Pemb(P)) % Peny(P),␣
 ↪Peny(P))



# AddPC : 2 pecahanc --> pecahanc
# AddPc(P1,P2) menjumlahkan pecahan campuran P1 dan P2, hasilnya adalah pecahan␣
 ↪campuran
#REALISASI dalam Python
def AddPC(P1,P2):
    return KonversiPtoPC(AddP(KonversiPCtoP(P1), KonversiPCtoP(P2)))

# SubPC : 2 pecahanc --> pecahanc
# SubPC(P1,P2) mengurangkan pecahan campuran P1 dengan P2, hasilnya adalah␣
 ↪pecahan campuran
#REALISASI dalam Python
def SubPC(P1,P2):
    return KonversiPtoPC(SubP(KonversiPCtoP(P1), KonversiPCtoP(P2)))

# MulPC : 2 pecahanc --> pecahanc
# MulPC(P1,P2) mengalikan pecahan campuran P1 dengan P2, hasilnya adalah␣
 ↪pecahan campuran
#REALISASI dalam Python
def MulPC(P1,P2):
    return KonversiPtoPC(MulP(KonversiPCtoP(P1), KonversiPCtoP(P2)))

# DivPC : 2 pecahanc --> pecahanc
# DivPC(P1,P2) membagi pecahan campuran P1 dengan P2, hasilnya adalah pecahan␣
 ↪campuran
#REALISASI dalam Python
def DivPC(P1,P2):
    return KonversiPtoPC(DivP(KonversiPCtoP(P1), KonversiPCtoP(P2)))

# RealPC : pecahanc --> real
# RealPC(P) mengubah pecahan campuran P ke dalam nilai real
#REALISASI dalam Python
def RealPC(P):
    if Bil(P) >= 0:
        return Bil(P) + (Pembc(P) / Penyc(P))
    else:
        return Bil(P) - (Pembc(P) / Penyc(P))
```

```
# CetakPC : pecahanc --> string
# CetakPC(P) mencetak nilai elemen pecahan campuran P ke layar
#REALISASI dalam Python
def CetakPC(P):
    print("Bilangan Bulat : ", Bil(P), "Pembilang: ", Pembc(P), "\nPenyebut: ",␣
 ↪Penyc(P))

#aplikasi
CetakPC(AddPC(pecahanc(2,3,4),pecahanc(3,1,2)))
CetakPC(SubPC(pecahanc(2,3,4),pecahanc(3,1,2)))
CetakPC(MulPC(pecahanc(2,3,4),pecahanc(3,1,2)))
CetakPC(DivPC(pecahanc(2,3,4),pecahanc(3,1,2)))
print(RealPC(pecahanc(2,3,4)))
```

```
Bilangan Bulat :  6 Pembilang:  2
Penyebut:  8
Bilangan Bulat :  0 Pembilang:  6
Penyebut:  8
Bilangan Bulat :  9 Pembilang:  5
Penyebut:  8
Bilangan Bulat :  0 Pembilang:  22
Penyebut:  28
2.75
```

[15]:
```
#EKIVALENSI DETIK(
#DEFINISI DAN SPESIFIKASI
#HHMMDD : integer[0..99999] --> <integer > 0, integer[0..23], integer[0..59],␣
 ↪integer[0..59]>
#HHMMDD(x) menghitung detik x ke dalam hari H, jam J, menit M dan detik D

#DEFINISI DAN SPESIFIKASI FUNGSI ANTARA
#QR : <integer > 0, integer > 0> --> <integer > 0, integer > 0>
#QR(N,D) menghasilkan <q,r> dengan q = N div D dan r = N mod D
#REALISASI dalam Python
def QR(N,D):
    return [N // D, N % D]

def HHMMDD(x):
    [H,sisaH] = QR(x,86400)
    [J,sisaJ] = QR(sisaH,3600)
    [M,D] = QR(sisaJ,60)

    return [H,J,M,D]

#APLIKASI
HHMMDD(10000)
```

[15]: [0, 2, 46, 40]

[7]:
```
-7 // 3
```

[7]: -3

[8]:
```
-7 % 3
```

[8]: 2

[6]:
```
5 % 3
```

[6]: 2

[ ]:
```

```