TUGAS MATA KULIAH SISTEM OPERASI PRAKTIK

Dosen : Iwan Hartadi Tri Untoro, S.T., M.Kom.

Asisten Dosen : Galang Aidil Akbar

“RESPONSI”

Oleh :

5200411144 Bella Primin

PROGRAM STUDI TEKNIK INFORMATIKA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS TEKNOLOGI YOGYAKARTA

2021

1. Membuat sebuah program yang mensimulasikan manajemen RAM didalam Komputer
   Pada program ini akan menjalankan sebuah perhitungan manajemen memory yang memiliki kriteria berupa nilai yang harus di inputkan untuk menjalankannya.
   Inputan tersebut berupa :
   - Kapasitas Total RAM
   - Kapasitas Total Petabit
   - Kapasitas RAM yang digunakan oleh Sistem Operasi
   - Kapasitas RAM yang digunakan oleh Program1
   - Kapasitas RAM yang digunakan oleh Program2

   Dan, memiliki output berupa:

   - Kapasitas Total RAM
   - Kapasitas Total Petabit
   - Kapasitas Total Per-petabit
   - Kapasitas RAM yang Terpakai
   - Kapasitas RAM yang Tidak Terpakai
   - Jumlah Blok yang Bernilai 1
   - Jumlah Blok yang Bernilai 0

   Seperti Code Program dibawah ini :

```
print("MANAJEMEN MEMORI")
print("================================")

RAM = float(input("Inputkan Kapasitas RAM (dalam MBps) \n => "))
Petabit = float(input("Inputkan Kapasitas Petabit (dalam MBps) \n => ")) #/1000
OS = float(input("Inputkan Kapasitas Sistem Operasi {RAM yang digunakan} (dalam MBps) \n => "))
KP1 = float(input("Inputkan Kapasitas Program 1 (dalam MBps) \n => "))
KP2 = float(input("Inputkan Kapasitas Program 2 (dalam MBps) \n => "))
print("")


#print("Kapasitas Total RAM: ")
KTRAM = RAM * 1024

#print("Kapasitas Total Petabit: ")
KTPetabit = OS *1024

#print("perhitungan TOTAL: ")
Total = KP1 + KP2

#print("perhitungan Per-Petabit: ")
PT = KTRAM / Petabit

#print("perhitungan RAM Terpakai: ")
RTP = KTPetabit + Total

#print("perhitungan RAM Tidak Terpakai: ")
RS = KTRAM - RTP

##print("perhitungan Blok 1: ")
```

```
##print("perhitungan Blok 1: ")
B1 = RTP / Petabit

##print("perhitungan Blok 0: ")
B0 = PT - B1


print("Kapasitas Total RAM \t\t\t: " + str(KTRAM) + " MBps.")
print("Kapasitas Total Petabit \t\t: " + str(Petabit) + " MBps.")
print("Kapasitas per Petabit \t\t\t: " + str(PT) + " MBps.")
print("Kapasitas RAM yang terpakai \t\t: " + str(RTP) + " MBps.")
print("Kapasitas RAM yang tidak terpakai \t: " + str(RS) + " MBps.")
print("Jumlah Blok yang bernilai 1 \t\t: " + str(B1) + " MBps.")
print("Jumlah Blok yang bernilai 0 \t\t: " + str(B0) + " MBps.")

if RT < 0:
    print("")
    print("!!! WARNING !!!")
    print("Kapasitas RAM tidak memungkinkan untuk menjalankan program!")

# KTRAM = Kapasitas RAM ---> Total RAM
# KTPetabit = Kapasitas Petabit ---> Total Petabit
# PT = Kapasitas Per Petabit
# OS = Kapasitas OS ---> Total RAM yang Terpakai
# RT = RAM tersisa ---> Total RAM yang Tidak Terpakai
# B1 = Jumlah Blok yang bernilai 1
# B0 = Jumlah Blok yang bernilai 0
```

Hasil :

```
MANAJEMEN MEMORI
================================
Inputkan Kapasitas RAM (dalam MBps)
 => 64
Inputkan Kapasitas Petabit (dalam MBps)
 => 16
Inputkan Kapasitas Sistem Operasi {RAM yang digunakan} (dalam MBps)
 => 4
Inputkan Kapasitas Program 1 (dalam MBps)
 => 2
Inputkan Kapasitas Program 2 (dalam MBps)
 => 2

Kapasitas Total RAM                     : 65536.0 MBps.
Kapasitas Total Petabit                 : 16.0 MBps.
Kapasitas per Petabit                   : 4096.0 MBps.
Kapasitas RAM yang terpakai             : 4100.0 MBps.
Kapasitas RAM yang tidak terpakai       : 61436.0 MBps.
Jumlah Blok yang bernilai 1             : 256.25 MBps.
Jumlah Blok yang bernilai 0             : 3839.75 MBps.
```

2. Membuat sebuah program yang mensimulasikan manajemen Penjadwalan dengan menggunakan algoritma Round Robin

   Pada program ini akan menjalankan sebuah perhitungan yang memiliki kriteria berupa nama program, dan lama proses pengerjaan serta Quantum time (Jatah Waktu) yang harus di inputkan untuk menjalankan program tersebut.

```
# Berfungsi untuk mencari waktu tunggu untuk semua proses
def WT(process, n, bt,wt, quantum):
    Ws = [0] * n
    for i in range(n):
        Ws[i] = bt[i]
    t = 0 # Waktu saat ini
    while(1):
        done = True
        # jalankan semua proses satu per satu berulang kali
        for i in range(n):
            # Jika waktu burst suatu proses lebih besar dari 0 maka hanya perlu diproses lebih lanjut
            if (Ws[i] > 0) :
                done = False # Ada proses di tunda
                if (Ws[i] > quantum) :
                    # Meningkatkan nilai t yaitu menunjukkan berapa lama suatu proses telah diproses
                    t += quantum
                    # Kurangi burst_time dari proses saat ini dengan kuantum
                    Ws[i] -= quantum
                # Jika waktu burst lebih kecil atau sama dengan kuantum. Siklus terakhir untuk proses ini
                else:
                    # Meningkatkan nilai t yaitu menunjukkan berapa lama suatu proses telah diproses
                    t = t + Ws[i]
                    # Waktu tunggu adalah waktu saat ini dikurangi waktu yang digunakan oleh proses ini
                    wt[i] = t - bt[i]
                    # Saat proses dijalankan sepenuhnya, buat waktu burst yang tersisa = 0
                    Ws[i] = 0
        # Jika semua proses sudah selesai
        if (done == True):
            break

def TT(process, n, bt, wt, tat):
    # Menghitung waktu penyelesaian
    for i in range(n):
        tat[i] = bt[i] + wt[i]

# Berfungsi untuk menghitung rata-rata waktu tunggu dan waktu putar balik.
def AWT(process, n, bt, quantum):
    wt = [0] * n
    tat = [0] * n
    # Berfungsi untuk mencari waktu tunggu semua proses
    WT(process, n, bt, wt, quantum)
    # Berfungsi untuk menemukan waktu putar balik untuk semua proses
    TT(process, n, bt, wt, tat)
    # Tampilkan proses beserta semua detailnya
    print(" ")
    print(" ")
    print("Processes    Burst Time     Waiting  Time    Turn-Around Time")
    total_wt = 0
    total_tat = 0
    for i in range(n):
        total_wt = total_wt + wt[i]
        total_tat = total_tat + tat[i]
        print(" ", i + 1, "\t\t", bt[i],
              "\t\t", wt[i], "\t\t", tat[i])
    print("\nRata Rata Waktu Tunggu = %.5f "%(total_wt /n) )
```

```
    print("\nRata Rata Waktu Tunggu = %.5f "%(total_wt /n) )
    print("Waktu penyelesaian rata-rata = %.5f "% (total_tat / n))


#FUNGSI UTAMA
print("ROUND ROBIN PROGRAM")
print(" ")

if __name__ =="__main__":
    # PROSSES
    pross = []
    n = int(input("masukan jumlah Manajemen Memori : "))
    for i in range(0, n):
      ele = int(input("Masukan Nilai Prosses : "))
      pross.append(ele)
    print(" ")
    # WAKTU BURST TIME/PENGERJAAN
    burst_time = []
    str(input("Nama Program : "))
    for i in range(0, n):
      ele = int(input("Masukan Waktu Pengerjaan : "))
      burst_time.append(ele)
    # Time quantum/Jatah Waktu
    quantum = int(input("Masukan jatah Waktu/Quantum time : "));
    AWT(pross, n, burst_time, quantum)
    print(" ")
```

Hasil :

```
    AWT(pross, n, burst_time, quantum)
    print(" ")
```

```
ROUND ROBIN PROGRAM

masukan jumlah Manajemen Memori : 5
Masukan Nilai Prosses : 20
Masukan Nilai Prosses : 25
Masukan Nilai Prosses : 35
Masukan Nilai Prosses : 30
Masukan Nilai Prosses : 40

Nama Program : PRIMIN PROGRAM
Masukan Waktu Pengerjaan : 10
Masukan Waktu Pengerjaan : 10
Masukan Waktu Pengerjaan : 10
Masukan Waktu Pengerjaan : 10
Masukan Waktu Pengerjaan : 10
Masukan jatah Waktu/Quantum time : 7


Processes    Burst Time     Waiting  Time    Turn-Around Time
  1             10             28             38
  2             10             31             41
  3             10             34             44
  4             10             37             47
  5             10             40             50

Rata Rata Waktu Tunggu = 34.00000
Waktu penyelesaian rata-rata = 44.00000
```