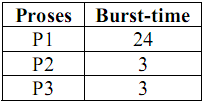
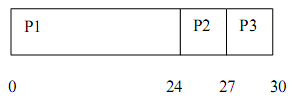
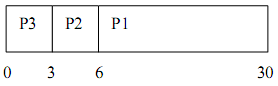
FIFO atau FCFS

* Contoh : ada 3 proses P1, P2, P3 dengan lama waktu kerja CPU (CPU Burst-time) masing-masing sbb :



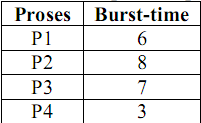


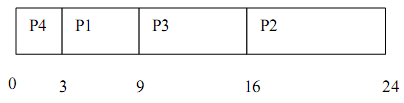
* Waktu tunggu P1 : 0 milidetik, P2 : 24, P3: 27
* Rata-rata waktu tunggu (Average Waiting Time / AWT) : (0+24+27)/3 = 17 milidetik
* Jika waktu kedatangan proses adalah P3, P2, P1 maka Gantt Chartnya adalah :



* AWT = (0+3+6)/3 **= 3 milidetik**

SJF





P1=3

P2=16

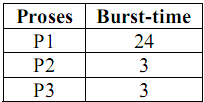
P3=9

P4=0

* AWT : (0+3+9+16) / 4**=7.0**

Round Robin

* Semua proses dianggap penting dan diberi sejumlah waktu pemroses yang disebut kwanta (quantum) atau time-slice dimana proses itu berjalan.



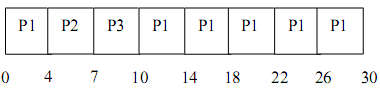
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Proses | Burst Time | Q=4 | Q=4 | Q-4 | Q=4 | Q=4 | Q=4 |
| P1 | 24 | 20 | 16 | 12 | 8 | 4 |  |
| P2 | 3<Q,+1 |  |  |  |  |  |  |
| P3 | 3<Q,+1 |  |  |  |  |  |  |

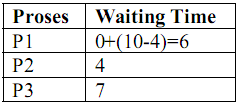
3<4 masa kuantum maka langsung dijumlah

3<4 masa kuantum maka langsung dijumlah

24-4 masa kuantum masih sisa 20, 16, 12, 8, 4

* Quantum 4 milidetik
  + P1 mendapat 4 milidetik pertama
  + 20 milidetik berikutnya akan disela P2 dan P3

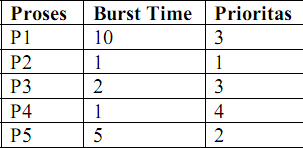


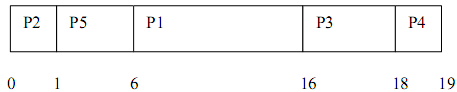


* AWT : (6+4+7)/**3 = 5,66 milidetik**

Prioritas(PS)

* ada 5 proses P1,P2,P3,P4,P5





* P1=6
* P2=0
* P3=16
* P4=18
* P5=1
* AWT = (6+0+16+18+1) **= 8,2 ms**

:LATIHAN

|  |  |  |
| --- | --- | --- |
| **Proses** | **Burst Time** | **Prioritas** |
| **P1** | 9 | 3 |
| **P2** | 4 | 4 |
| **P3** | 3 | 1 |
| **P4** | 8 | 2 |
| **P5** | 15 | 5 |
| **P6** | 5 | 6 |

Gambarkan dan hitunglah waktu rata-rata untuk waktu tunggu rata-rata masing-masing proses gambarkan *gantt chart* waktu tunggu masing-masing proses, AWT ***(Average Waiting Time)*** dengan menggunakan algoritma penjadwalan :

* 1. *FIFO (Fist In Firt Out) atau FCFS (First Come First Serve)*
  2. *SJF (Shortest Job First)*
  3. *RR (Round Robin) quantum 4*
  4. *PS(Priorty Scheduling)*

Jawab :

* 1. FIFO

|  |  |
| --- | --- |
| **Proses** | **Burst Time** |
| **P1** | 9 |
| **P2** | 4 |
| **P3** | 3 |
| **P4** | 8 |
| **P5** | 15 |
| **P6** | 5 |

Gantt chart

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P4 | P5 | P6 |
| 0 9 | 13 | 16 | 24 | 39 | 44 |

Waktu tunggu

P1=0,

P2=9,

P3=13,

P4=16,

P5=24,

P6=39

AWT=101/6

AWT=**16.8**

* 1. SJF (Shortest Job First)

|  |  |  |
| --- | --- | --- |
| **Proses** | **Burst Time** | **Short** |
| **P1** | 9 | 5 |
| **P2** | 4 | 2 |
| **P3** | 3 | 1 |
| **P4** | 8 | 4 |
| **P5** | 15 | 6 |
| **P6** | 5 | 3 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| P3 | P2 | P6 | P4 | P1 | P5 |  |
| 0 3 | 7 | 12 | 20 | 29 | 44 |  |

P1=20

P2=3

P3=0

P4=12

P5=29

P6=7

AWT =109/6

AWT =**11,83**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Proses** | **Burst Time** | **Short** | **Q=4** | **Q=4** | **Q=4** |
| **P1** | 9 | 5 | 5 | 1<=q,+ |  |
| **P2** | 4 | 2 |  |  |  |
| **P3** | 3<=q,+ | 1 |  |  |  |
| **P4** | 8 | 4 | 4 |  |  |
| **P5** | 15 | 6 | 11 | 7 | 3<=q,+ |
| **P6** | 5 | 3 | 1<=q,+ |  |  |

* 1. RR

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P4 | P5 | P6 | P1 | P4 | P5 | P6 | P1 | P5 | P5 |
| 0 4 | 8 | 11 | 15 | 19 | 23 | 27 | 31 | 35 | 36 | 37 | 41 | 44 |

**Quantum = 4**

P1=0+(23-4)+(36-27)=19+9=28

P2=4

P3=8

P4=11+(27-15)=11+12=23

P5=15+(31-19)+(37--35)+(41-41)=15+12+2=29

P6=19+(35-23)=19+12=31

(28+4+8+23+29+31)/6=**20.5**

* 1. PS

|  |  |  |
| --- | --- | --- |
| **Proses** | **Burst Time** | **Prioritas** |
| **P1** | 9 | 3 |
| **P2** | 4 | 4 |
| **P3** | 3 | 1 |
| **P4** | 8 | 2 |
| **P5** | 15 | 5 |
| **P6** | 5 | 6 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P3 | P4 | P1 | P2 | P5 | P6 |
| 0 3 | 11 | 20 | 24 | 39 | 44 |

P1=11

P2=20

P3=0

P4=3

P5=24

P6=39

AWT=97/6

**AWT=16,16**

**CONTOH**

Langsung saja ya sahabat-sahabatku...

Diketahui ada 4 proses yang arrival time dan burst time sebagai berikut.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Proses** | **Arrival Time** | **Burst Time** | **Q=3** | **Q=3** |
| P1 | 0 | 8 | 5 | 2<=Q,+ |
| P2 | 3 | 5 | 2<=Q,+ |  |
| P3 | 5 | 2<=Q,+ |  |  |
| P4 | 6 | 6 | 3<=Q,+ |  |

Hitung Waiting Time dan Average Time menggunakan metode penjadwalan Penjadwalan Round Robin.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P4 | P1 | P2 | P4 | P1 |
| 0 3 | 6 | 8 | 11 | 14 | 16 | 19 | 21 |

P1=0+(11-3)+(19-14)=13

P2=3+(14-6)=11

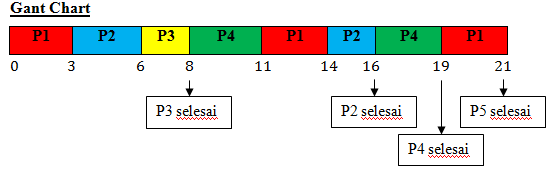
P3=6

P4=8+(16-11)=13

13+11+6+13=10.75

**Jawabannya sebagai berikut.**

Metode Round Robin yaitu metode yang setiap prosesnya memperoleh alokasi waktu CPU dalam quantum waktu yang telah diketahui. Disni quantum time diketahui 3 ms.

[](http://1.bp.blogspot.com/-l9USNRB0Muo/UremDP-RkmI/AAAAAAAAAaM/5aOKrlmFKC8/s1600/g4.png)

[Cara menghitung round robin](https://davidimann.blogspot.com/2019/02/cara-menghitung-round-robin.html)

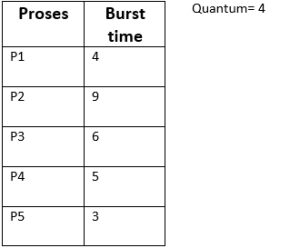
**Round Robin** merupakan salah satu penjadualan proses, dimana algoritma ini menggilir proses yang ada di antrian. Proses akan mendapatkan jatah sebesar time quantum. Jika time quantumnya habis atau proses sudah selesai, CPU akan dialokasikan ke proses berikutnya. Pada penjadwalan proses ini, tidak ada proses yang diprioritaskan, semua proses mendapatkan qr waktu yang sama dari CPU.

Contoh :

**• Langkah Pertama :**

Membuat Gantt Chart

Masukkan proses yang berurutan, kemudian kurangi Burst Time dengan nilai quantum.

[](https://bbloodabsurdkone.files.wordpress.com/2014/12/rr2.png)

P1 :

Burst time – Quantum

4 – 4 = 0.

Waktu = 4

P2:

Burst time – Quantum

9 – 4 = 5.

Sisa = 5

Waktu = 4

P3:

Burst time – Quantum

6 – 4 = 2.

Sisa = 2

Waktu = 4

P4:

Burst time – Quantum

5 – 4 = 1.

Sisa = 1

Waktu antrian = 4

P5:

Burst time – Quantum

3 – 4 = (-1).

Waktu antrian = 3

Proses yang memiliki sisa, kemudian dimasukkan kembali kedalam antrian sesuai dengan urutan prosesnya.

P2:

Sisa Burst time – Quantum

5 – 4 = 1

Sisa = 1

Waktu antrian = 4

P3:

Sisa Burst time – Quantum

2 – 4 = (-2)

Waktu antrian = 2

P4:

Sisa Burst time – Quantum

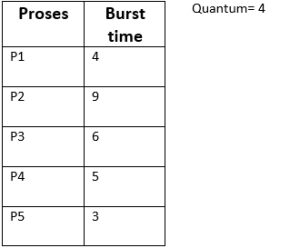
1 – 4 = (-3)

Waktu antrian = 1

P2:

Sisa Burst time – Quantum

1 – 4 = (-3)

[](https://bbloodabsurdkone.files.wordpress.com/2014/12/rr2.png)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Proses | Burst Time | Q=4 | Q=4 | Q=4 |
| P1 | 4 |  |  |  |
| P2 | 9 | 5 | 1<=q, + |  |
| P3 | 6 | 2<=q,+ |  |  |
| P4 | 5 | 1<=q,+ |  |  |
| P5 | 3<=q,+ |  |  |  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| P1 | P2 | P3 | P4 | P5 | P2 | P3 | P4 | P2 |
| 0 4 | 8 | 12 | 16 | 19 | 23 | 25 | 26 | 27 |

P1=0

P2=4+(19-8)+(26-23)

P3=8+(23-12)

P4=12+(25-16)

P5=16

(0 + 18 + 19 + 21 + 16)/ 5

= 74 / 5

**= 14,8**

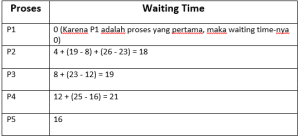
Waktu antrian = 1

Setelah selesai menentukan waktu antrian, kemudian buat tabel seperti dibawah ini

[RR3](https://bbloodabsurdkone.files.wordpress.com/2014/12/rr3.png)

**• Langkah Kedua :**

Menghitung AWT (Average Waiting Time)

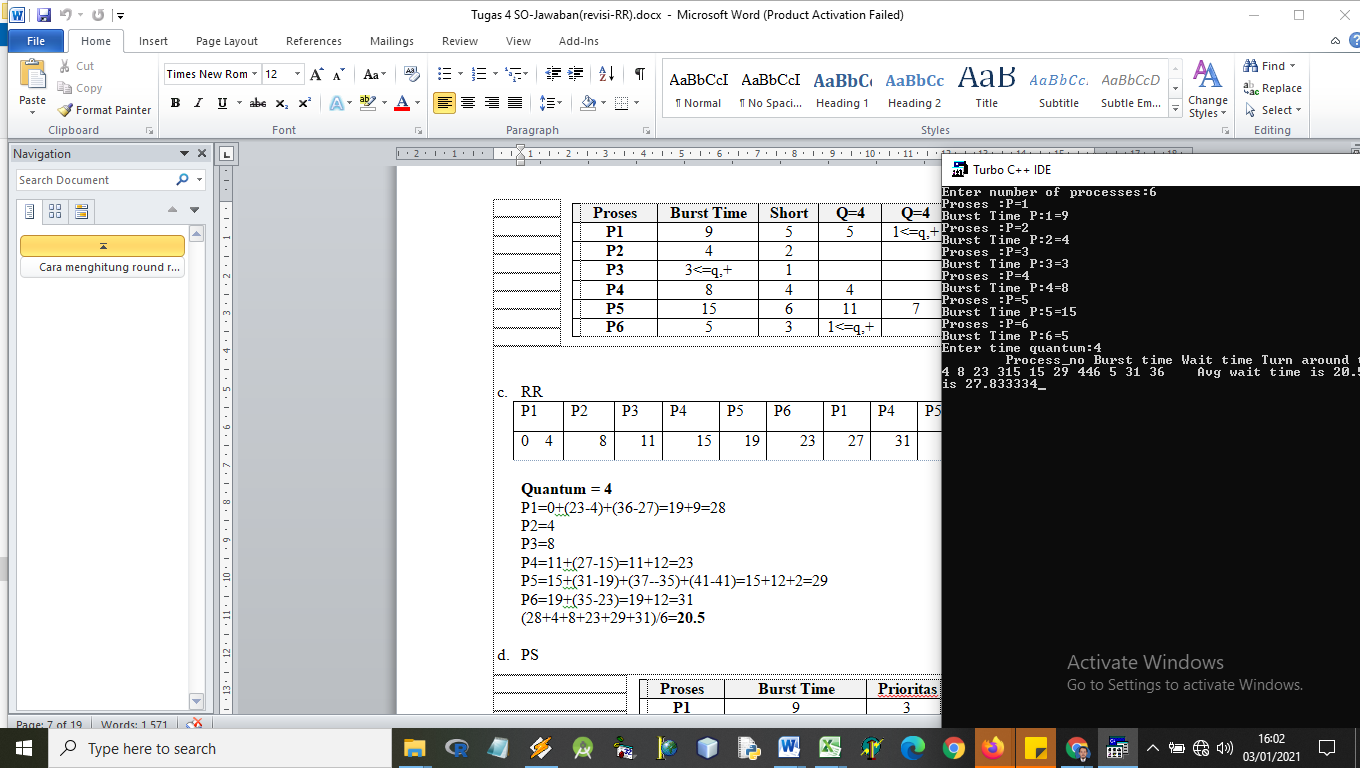
**[](https://bbloodabsurdkone.files.wordpress.com/2014/12/rr4.png)**

AWT yang terjadi adalah:

(0 + 18 + 19 + 21 + 16)/ 5

= 74 / 5

= 14,8



**//Algoritma FIFO/FCFS**

**#include<stdio.h>**

**#include<string.h>**

**main()**

**{**

**int n, ar[100], b[100], i, j, tmp, wt[100], ta[100], time[100];**

**int totWT=0, totTA=0;**

**float AvWT, AvTA;**

**char name[20][20], tmpName[20];**

**printf("Program Penjadwalan CPU FCFS");**

**puts("");**

**printf("Banyak Proses\t= "); scanf("%d",&n);**

**puts("");**

**// Masukkan data yang diproses**

**for(i=0; i<n; i++){**

**fflush(stdin);**

**printf("Nama Proses\t= "); gets(name[i]);**

**printf("Arrival time\t= "); scanf("%d", &ar[i]);**

**printf("Burst time\t= "); scanf("%d", &b[i]);**

**puts("");**

**}**

**// SORTING Data**

**for(i=0; i<n; i++){**

**for(j=i+1; j<n; j++)**

**if(ar[i]>ar[j]){**

**//tukar nama**

**strcpy(tmpName, name[i]);**

**strcpy(name[i], name[j]);**

**strcpy(name[j], tmpName);**

**//tukar arrival time**

**tmp=ar[i];**

**ar[i]=ar[j];**

**ar[j]=tmp;**

**//tukar burst**

**tmp=b[i];**

**b[i]=b[j];**

**b[j]=tmp;**

**}**

**}**

**time[0]=ar[0];**

**puts("\n\t.:: Process Table ::.");**

**puts("==========================================");**

**printf("| no | proses\t | time arrival\t | burst |\n");**

**puts("=========================================");**

**for (i=0; i<n; i++){**

**printf("| %2d | %s\t | \t%d\t | %d\t |\n", i+1, name[i], ar[i], b[i]);**

**time[i+1]=time[i]+b[i]; //menghitung time pada ganchart**

**wt[i]=time[i]-ar[i];**

**ta[i]=time[i+1]-ar[i];**

**totWT+=wt[i];**

**totTA+=ta[i];**

**}**

**puts("==========================================");**

**printf("\tTotal waiting time\t= %d \n", totWT);**

**printf("\tTurn arround time\t= %d \n", totTA);**

**puts("\n\t.:: Time Process Table ::.");**

**puts("==================================================");**

**printf("| no | proses\t | waiting time\t | turn arround\t |\n");**

**puts("====================================================");**

**for(i=0; i<n; i++){**

**printf("| %2d | %s\t | \t%d\t | \t%d\t |\n", i+1, name[i], wt[i], ta[i]);**

**}**

**puts("==================================================");**

**//untuk Gant Chart**

**puts("");**

**puts("\t.:: Gant-Chart ::.\n");**

**for(i=0; i<n; i++){**

**printf("%s\t ", name[i]);**

**}**

**puts("");**

**for(i=0; i<n; i++){**

**printf("|=========");**

**}**

**printf("|\n");**

**for(i=0; i<=n; i++){**

**printf(" %d\t ", time[i]);**

**}**

**printf("\tdiperoleh dari penjumlahan Burst\n");**

**puts("");**

**AvWT=(float)totWT/n; //average waiting time**

**AvTA=(float)totTA/n; //average turn arround time**

**printf("\tAverage Waiting Time : %f\n", AvWT);**

**printf("\tAverage Turn Arround TIme : %f\n", AvTA);**

**}**

**//Algoritma SJF (shortest Job Firt)**

**#include<stdio.h>**

**#include<string.h>**

**void main()**

**{**

**int i,j,n,brust\_time[10],start\_time[10],end\_time[10],wait\_time[10],temp,tot;**

**float avg;**

**printf("Enter the No. of jobs:\n\n");**

**scanf("%d",&n);**

**for(i=1;i<=n;i++)**

**{**

**printf("\n \n Enter %d process burst time:\n",i);**

**scanf("%d",&brust\_time[i]);**

**}**

**for(i=1;i<=n;i++)**

**{**

**for(j=i+1;j<=n;j++)**

**{**

**if(brust\_time[i]>brust\_time[j])**

**{**

**temp=brust\_time[i];**

**brust\_time[i]=brust\_time[j];**

**brust\_time[j]=temp;**

**}**

**}**

**if(i==1)**

**{**

**start\_time[1]=0;**

**end\_time[1]=brust\_time[1];**

**wait\_time[1]=0;**

**}**

**else**

**{**

**start\_time[i]=end\_time[i-1];**

**end\_time[i]=start\_time[i]+brust\_time[i];**

**wait\_time[i]=start\_time[i];**

**}**

**}**

**printf("\n\n BURST TIME \t STARTING TIME \t END TIME \t WAIT TIME\n");**

**printf("\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");**

**for(i=1;i<=n;i++)**

**{**

**printf("\n %5d %15d %15d %15d",brust\_time[i],start\_time[i],end\_time[i],wait\_time[i]);**

**}**

**printf("\n \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");**

**for(i=1,tot=0;i<=n;i++)**

**tot+=wait\_time[i];**

**avg=(float)tot/n;**

**printf("\n\n\n AVERAGE WAITING TIME=%f",avg);**

**for(i=1,tot=0;i<=n;i++)**

**tot+=end\_time[i];**

**avg=(float)tot/n;**

**printf("\n\n AVERAGE TURNAROUND TIME=%f",avg);**

**for(i=1,tot=0;i<=n;i++)**

**tot+=start\_time[i];**

**avg=(float)tot/n;**

**printf("\n\n AVERAGE RESPONSE TIME=%f\n\n",avg);**

**}**

**//ROUND ROBIN**

**#include<stdio.h>**

**#include<conio.h>**

**main()**

**{**

**int st[10],bt[10],wt[10],tat[10],n,tq;**

**int i,count=0,swt=0,stat=0,temp,sq=0;**

**float awt=0.0,atat=0.0;**

**clrscr();**

**printf("Enter number of processes:");**

**scanf("%d",&n);**

**//printf("Enter burst time for sequences:");**

**for(i=0;i<n;i++)**

**{**

**printf("Proses :P=");scanf("%d",&st[i]);**

**printf("Burst Time P:%d=",i+1);scanf("%d",&bt[i]);**

**st[i]=bt[i];**

**}**

**printf("Enter time quantum:");**

**scanf("%d",&tq);**

**while(1)**

**{**

**for(i=0,count=0;i<n;i++)**

**{**

**temp=tq;**

**if(st[i]==0)**

**{**

**count++;**

**continue;**

**}**

**if(st[i]>tq)**

**st[i]=st[i]-tq;**

**else**

**if(st[i]>=0)**

**{**

**temp=st[i];**

**st[i]=0;**

**}**

**sq=sq+temp;**

**tat[i]=sq;**

**}**

**if(n==count)**

**break;**

**}**

**for(i=0;i<n;i++)**

**{**

**wt[i]=tat[i]-bt[i];**

**swt=swt+wt[i];**

**stat=stat+tat[i];**

**}**

**awt=(float)swt/n;**

**atat=(float)stat/n;**

**printf("\tProcess\_no Burst time Wait time Turn around time");**

**for(i=0;i<n;i++)**

**printf("%d %d %d %d",i+1,bt[i],wt[i],tat[i]);**

**printf("\tAvg wait time is %f Avg turn around time is %f",awt,atat);**

**getch();**

**}**

**//priority scheduling**

**#include<stdio.h>**

**#include<string.h>**

**main()**

**{**

**int n, ar[100], b[100], i, j, tmp, wt[100], ta[100], time[100];**

**int totWT=0, totTA=0;**

**float AvWT, AvTA;**

**char name[20][20], tmpName[20];**

**printf("Program Penjadwalan Priority Scheduling");**

**puts("");**

**printf("Banyak Proses\t= "); scanf("%d",&n);**

**puts("");**

**// Masukkan data yang diproses**

**for(i=0; i<n; i++){**

**fflush(stdin);**

**printf("Nama Proses\t= "); gets(name[i]);**

**//printf("Arrival time=0\t= "); scanf("%d", &ar[i]);**

**printf("Burst time\t= "); scanf("%d", &b[i]);**

**puts("");**

**}**

**// SORTING Data**

**for(i=0; i<n; i++){**

**for(j=i+1; j<n; j++)**

**if(ar[i]>ar[j]){**

**//tukar nama**

**strcpy(tmpName, name[i]);**

**strcpy(name[i], name[j]);**

**strcpy(name[j], tmpName);**

**//tukar arrival time**

**tmp=ar[i];**

**ar[i]=ar[j];**

**ar[j]=tmp;**

**//tukar burst**

**tmp=b[i];**

**b[i]=b[j];**

**b[j]=tmp;**

**}**

**}**

**time[0]=ar[0];**

**puts("\n\t.:: Process Table ::.");**

**puts("==========================================");**

**printf("| no | proses\t | time arrival\t | burst |\n");**

**puts("=========================================");**

**for (i=0; i<n; i++){**

**printf("| %2d | %s\t | \t%d\t | %d\t |\n", i+1, name[i], ar[i], b[i]);**

**time[i+1]=time[i]+b[i]; //menghitung time pada ganchart**

**wt[i]=time[i]-ar[i];**

**ta[i]=time[i+1]-ar[i];**

**totWT+=wt[i];**

**totTA+=ta[i];**

**}**

**puts("==========================================");**

**printf("\tTotal waiting time\t= %d \n", totWT);**

**printf("\tTurn arround time\t= %d \n", totTA);**

**puts("\n\t.:: Time Process Table ::.");**

**puts("==================================================");**

**printf("| no | proses\t | waiting time\t | turn arround\t |\n");**

**puts("====================================================");**

**for(i=0; i<n; i++){**

**printf("| %2d | %s\t | \t%d\t | \t%d\t |\n", i+1, name[i], wt[i], ta[i]);**

**}**

**puts("==================================================");**

**//untuk Gant Chart**

**puts("");**

**puts("\t.:: Gant-Chart ::.\n");**

**for(i=0; i<n; i++){**

**printf("%s\t ", name[i]);**

**}**

**puts("");**

**for(i=0; i<n; i++){**

**printf("|=========");**

**}**

**printf("|\n");**

**for(i=0; i<=n; i++){**

**printf(" %d\t ", time[i]);**

**}**

**printf("\tdiperoleh dari penjumlahan Burst\n");**

**puts("");**

**AvWT=(float)totWT/n; //average waiting time**

**AvTA=(float)totTA/n; //average turn arround time**

**printf("\tAverage Waiting Time : %f\n", AvWT);**

**printf("\tAverage Turn Arround TIme : %f\n", AvTA);**

**}**