**LAPORAN**

**[CPU Scheduling] Kode Program Round Robin**



Disusun Oleh:

Aldow Fan Dzikri (NRP. 3121500022)

1 D3 ITA

Dosen Pengampu:

Fitri Setyorini ST. M.Sc

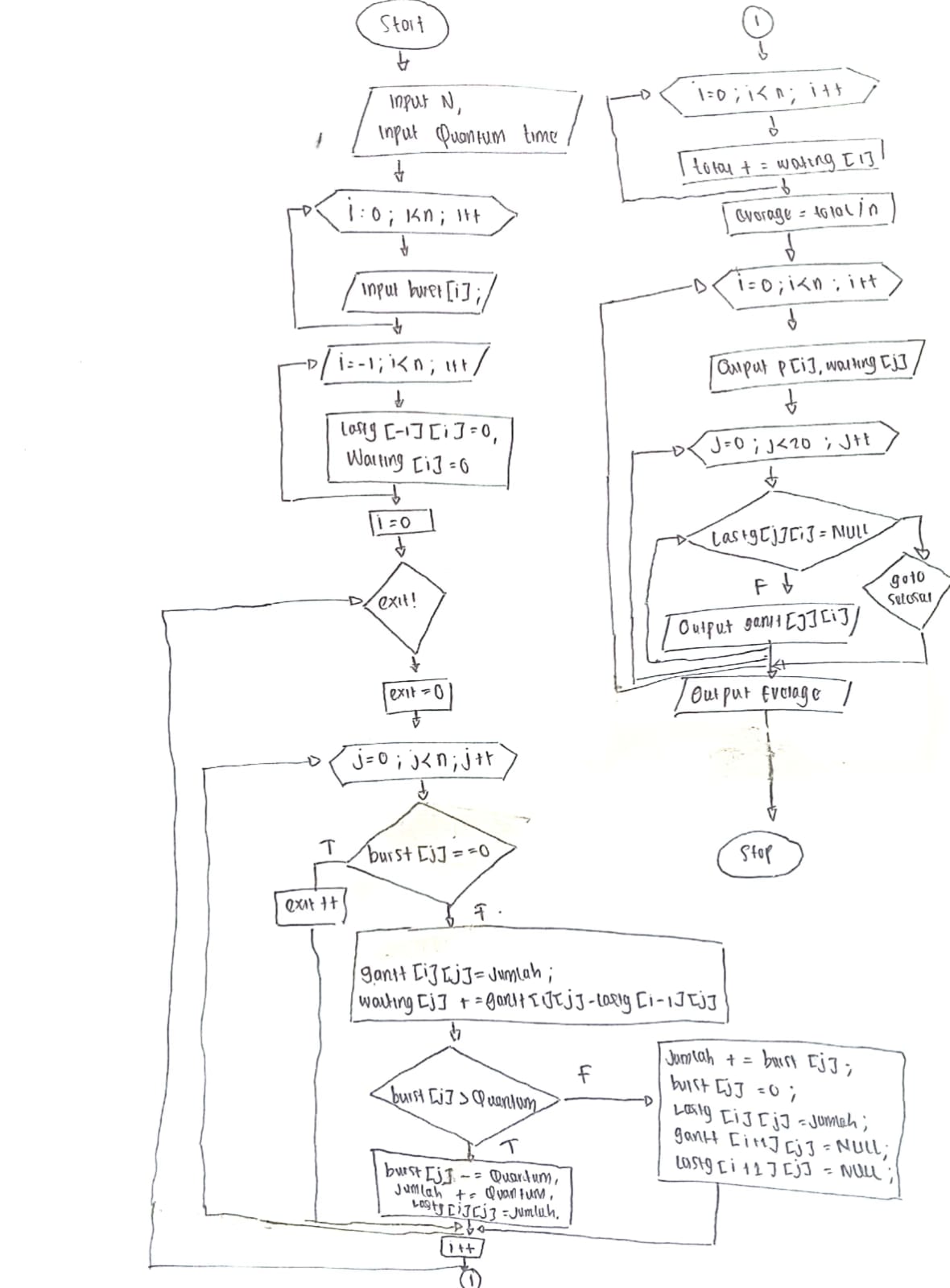
**PROGRAM STUDI D3 TEKNIK INFORMATIKA**

**DEPARTEMEN TEKNIK INFORMATIKA DAN KOMPUTER**

**POLITEKNIK ELEKTRONIKA NEGERI SURABAYA**

**2022**

**FLowchart:**

****

**Source Code:**

#include <stdio.h>

#include <stdlib.h>

main()

{

*int* exit; *// Keluar looping while*

*int* n, i, j, burst[20], quantum, gantt[20][20], lastg[20][20], waiting[20];

*int* jumlah = 0; *// Sebagai index burst untuk mencari gannt chart*

*float* total = 0, everage; *// Average waiting time*

    printf("Masukkan banyak proses (MAX 20): "); fflush(stdin); scanf("%i", &n);

    printf("Masukkan time quantum : "); fflush(stdin); scanf("%i", &quantum);

    for(i=0;i<n;i++) *// Input*

    {

        printf("Burst time P[%i] : ", i); fflush(stdin); scanf("%i", &burst[i]);

    }

    for(i=-1;i<n;i++) *// Inisialisasi pertama kali*

    {

        lastg[-1][i] = 0;

        waiting[i] = 0;

    }

    i = 0;

    while(exit != n) *// Keluar jika sudah habis semua*

    {

        exit = 0;

        for(j=0;j<n;j++)

        {

            if(burst[j] == 0){

                exit++;

            }

            else {

                gantt[i][j] = jumlah;

                waiting[j] += gantt[i][j] - lastg[i-1][j];

                if(burst[j] > quantum){

                    burst[j] -= quantum;

                    jumlah += quantum;

                    lastg[i][j] = jumlah;

                }

                else {

                    jumlah += burst[j];

                    burst[j] = 0;

                    lastg[i][j] = jumlah;

                    gantt[i+1][j] = NULL;

                    lastg[i+1][j] = NULL;

                }

            }

        }

        i++;

    }

    for(i=0;i<n;i++) *// Menghitung everage*

    {

        total += waiting[i];

    }

    everage = total / n;

    printf("\nProses\tWaiting time\tGantt chart\n"); *// Output*

    printf("===============================================\n");

    for(i=0;i<n;i++)

    {

        printf("P[%i]\t%i\t\t", i, waiting[i]);

        for(j=0;j<20;j++)

        {

            if(lastg[j][i] == NULL){

                goto selesai;

            }

            else {

                printf("%i-%i ", gantt[j][i], lastg[j][i]);

            }

        }

        selesai :

        printf("\n");

    }

    printf("===============================================\n");

    printf("Average waiting time = %.2f\n", everage);

}

Berikut ini adalah penerpan source code diatas yang menghasilkan berupa output penyelesaian soal dari sebuah kasus yang ada di bawah ini:

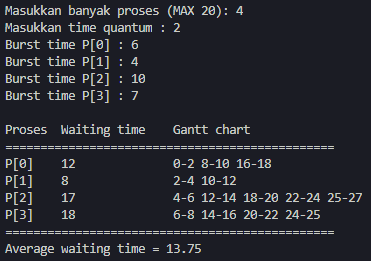
• Jika ada 4 proses, P1-P4 datang bersama-sama pada detik ke 0 dan

Burst time berurutan 6,4,10,7

• Time quantum 2

• Hitung Gant Chart, Waiting Time dan Average Waiting Time dengan

Round Robin



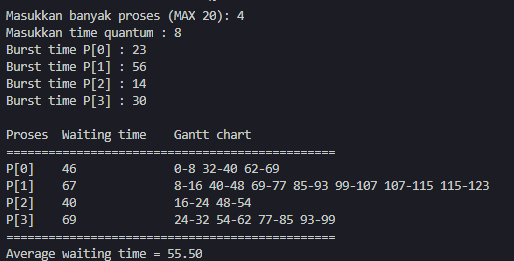
• Jika ada 4 proses, P1-P4 datang bersama-sama pada detik ke 0 dan

Burst time berurutan 23, 56, 14, 30

• Time quantum 8

• Hitung Gant Chart, Waiting Time dan Average Waiting Time dengan

Round Robi



Kerjakan soal berikut dengan Round Robin. Diketahui ada 4 proses, P1-P4. Semua proses datang pada detik 0. Masing-masing memiliki burst time : 70,50,65,48, Time quantum yang digunakan adalah 20. (a) Buatlah gantt chart dari keempat proses (b) Hitunglah waiting time tiap proses (c) Hitunglah average waiting time untuk keseluruhan proses

