ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA KHOA HỌC VÀ KỸ THUẬT THÔNG TIN

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

HỆ ĐIỀU HÀNH
IT007.O21.LT

BÁO CÁO

BÁO CÁO THỰC HÀNH LAB 4

Sinh viên thực hiện:
Trần Duy Tân - 22550020

Giảng viên:
ThS. Đỗ Trí Nhựt

Thành phố Hồ Chí Minh, năm 2024

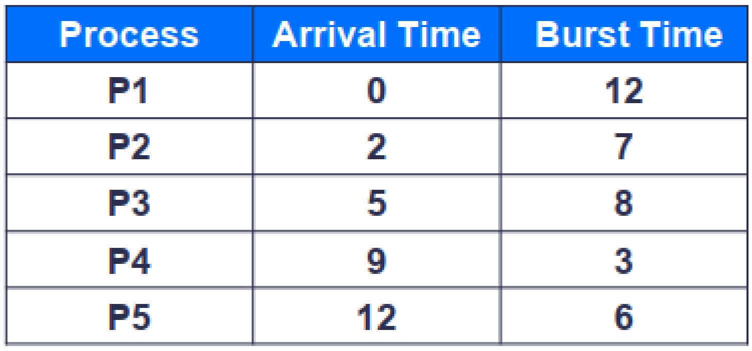
1. Viết chương trình mô phỏng giải thuật SJF với các yêu cầu sau:

❖ Nhập số lượng process

❖ Nhập process name, arrival time, burst time

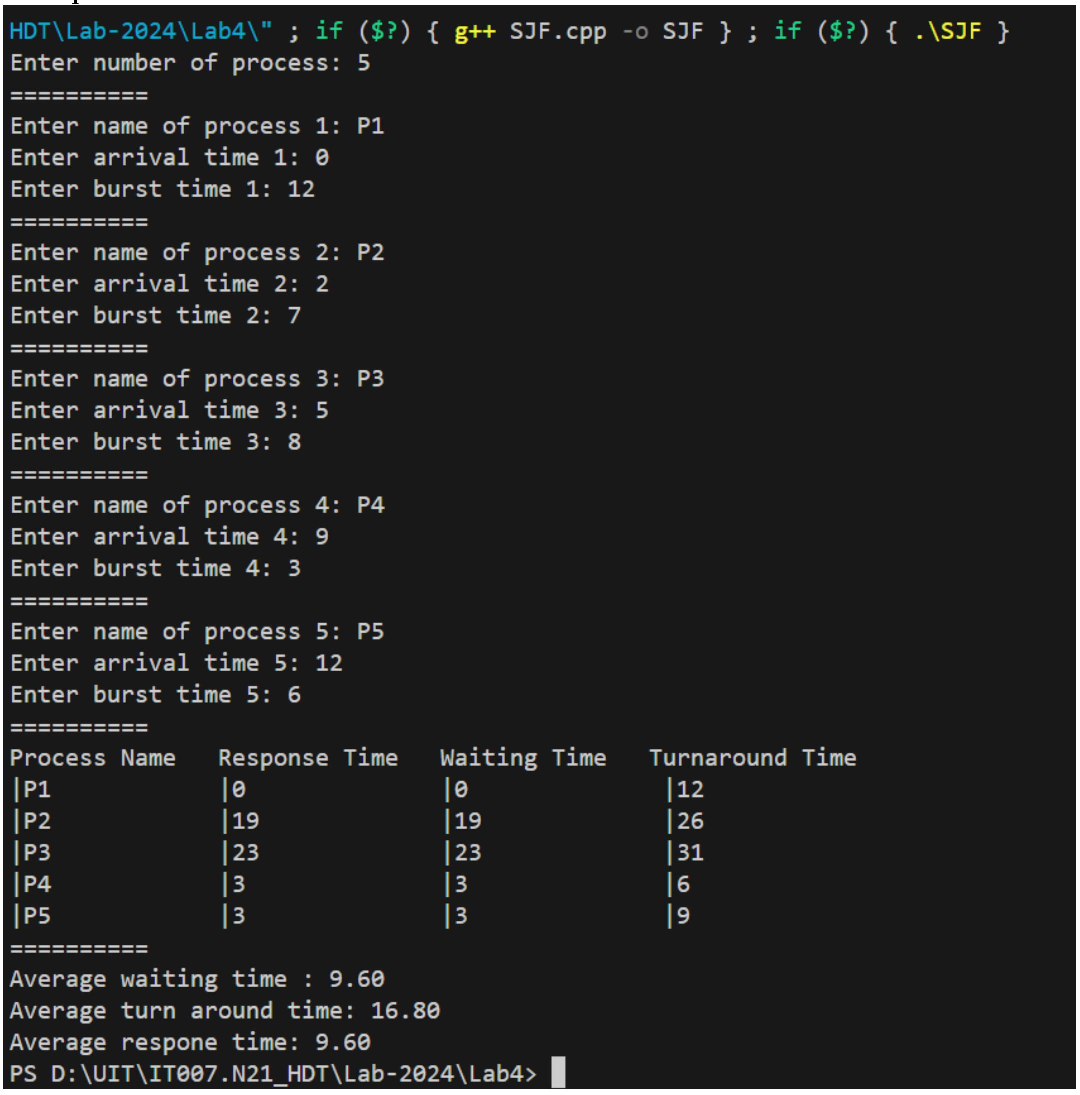
❖ In ra Process name, response time, waiting time, turnaround time, average waiting time, average turnaround time.

Test Case



Process	Arrival Time	Burst Time
P1	0	12
P2	2	7
P3	5	8
P4	9	3
P5	12	6

Kết quả:



```
HDT\Lab-2024\Lab4\" ; if ($?) { g++ SJF.cpp -o SJF } ; if ($?) { .\SJF }
Enter number of process: 5
==========
Enter name of process 1: P1
Enter arrival time 1: 0
Enter burst time 1: 12
==========
Enter name of process 2: P2
Enter arrival time 2: 2
Enter burst time 2: 7
==========
Enter name of process 3: P3
Enter arrival time 3: 5
Enter burst time 3: 8
==========
Enter name of process 4: P4
Enter arrival time 4: 9
Enter burst time 4: 3
==========
Enter name of process 5: P5
Enter arrival time 5: 12
Enter burst time 5: 6
==========
Process Name   Response Time   Waiting Time   Turnaround Time
|P1            |0              |0              |12
|P2            |19             |19             |26
|P3            |23             |23             |31
|P4            |3              |3              |6
|P5            |3              |3              |9
==========
Average waiting time : 9.60
Average turn around time: 16.80
Average respone time: 9.60
PS D:\UIT\IT007.N21_HDT\Lab-2024\Lab4>
```

Code demo

```c
#include <stdio.h>
#include <stdlib.h>
#define N 10010
#define INF 1e9

struct process {
   char name[15];
   int id;
   float arr, brust, finish, wait_time, respones_time, turnround_time;
};
int num_process;

void swap(struct process *a, struct process *b) {
   struct process tmp = *a;
   *a = *b;
   *b = tmp;
}

void sort_SJF(struct process P[]) {
   for (int i = 0; i < num_process - 1; i++)
       for (int j = i + 1; j < num_process; j++)
           if (P[j].arr < P[i].arr) swap(&P[j], &P[i]);
           else if (P[i].arr == P[j].arr && P[j].brust < P[i].brust)
swap(&P[j], &P[i]);
}

void output(struct process P[]) {
   // sort increasing id
   for (int i = 0; i < num_process - 1; i++)
       for (int j = i + 1; j < num_process; j++)
           if (P[j].id < P[i].id) swap(&P[i], &P[j]);

   printf("Process Name   Response Time   Waiting Time   Turnaround Time\n");

   float avg_waiting_time = 0;
   float avg_turnaround = 0;
   float avg_respone_time = 0;
   for (int i = 0; i < num_process; i++) {
       printf("|%-14s|%-15d|%-15d|%d\n", P[i].name, (int) P[i].respones_time,
(int) P[i].wait_time,
              (int) P[i].turnround_time);

       avg_waiting_time += P[i].wait_time;
       avg_turnaround += P[i].turnround_time;
       avg_respone_time += P[i].respones_time;
   }
   avg_waiting_time /= num_process;
   avg_turnaround /= num_process;
   avg_respone_time /= num_process;

   printf("==========\n");
   printf("Average waiting time : %.2f\n", avg_waiting_time);
   printf("Average turn around time: %.2f\n", avg_turnaround);
   printf("Average respone time: %.2f\n", avg_respone_time);
}


void SJF(struct process P[]) {
   int *done = (int *) malloc(num_process * sizeof(int));
   for (int i = 0; i < num_process; i++) done[i] = 0;

   float time_current = P[0].arr + P[0].brust;
   P[0].finish = time_current;
   P[0].turnround_time = P[0].finish - P[0].arr;
   P[0].wait_time = P[0].turnround_time - P[0].brust;
   P[0].respones_time = P[0].wait_time;

   done[0] = 1;

   for (int i = 1; i < num_process; i++) {
       float shortestBrust = INF;
       int idx = -1;

       for (int j = 0; j < num_process; j++) {
           if (!done[j] && time_current >= P[j].arr && shortestBrust >
P[j].brust) {
               shortestBrust = P[j].brust;
               idx = j;
           }
       }

       if (idx == -1) {
           float shortestarr = INF;
           for (int j = 0; j < num_process; j++)
               if (!done[j] && shortestarr > P[j].arr && P[j].arr >
time_current) {
                   shortestarr = P[j].arr;
                   idx = j;
               }
           if (idx == -1) break;
           time_current = shortestarr;
       }

       time_current += P[idx].brust;
```

```
        P[idx].finish = time_current;
        P[idx].turnround_time = P[idx].finish - P[idx].arr;
        P[idx].wait_time = P[idx].turnround_time - P[idx].brust;
        P[idx].respones_time = P[idx].wait_time;
        done[idx] = 1;
    }
}

int main() {
    struct process P[N];
    printf("Enter number of process: ");
    scanf("%d", &num_process);
    for (int i = 0; i < num_process; i++) {
        printf("==========\n");
        printf("Enter name of process %d: ", (i + 1));
        scanf("%s", &P[i].name);
        printf("Enter arrival time %d: ", (i + 1));
        scanf("%f", &P[i].arr);
        printf("Enter burst time %d: ", (i + 1));
        scanf("%f", &P[i].brust);
        P[i].finish = P[i].turnround_time = P[i].wait_time = P[i].respones_time
= 0;
        P[i].id = i;
    }
    printf("==========\n");
    sort_SJF(P);

    SJF(P);
    output(P);
    return 0;
}
```