MULTIPROCESSOR AND REAL-TIME SCHEDULING SHORTEST-JOB-FIRST (SJF) SCHEDULING ALGORITHM

Adel Abdullah Abbas

Department of Computer Science
Ministry Of Higher Education & Scientific Research Baghdad, Iraq

ABSTRACT: *Program that was designed to explains the application of the SJF(shortest –job-first) The purpose is to enter 4 CPU different values and implemented in a certain time in addition to sequencing the execution of these processors from slowest to fastest, and thus calculate the time for each processor, giving average the total time to take actions all 4 operations are complete and correct and the time of the execution of each Processor through the sequence to respond to the execution of the work and get results in the (special code) form output screen of my next explanation and used language visual basic 6.0 in the design of this program.*

KEYWORDS: CPU scheduling, Algorithms, Shortest Job First.

INTRODUCTION

This algorithm associates with each process the length of its next CPU burst and it take the shortest next CPU burst is chosen and big effect on short jobs, small effect on long, better avg in response time and the other name of this algorithm is Shortest-Process-Next (SPN) Shortest-Job-First (SJF) is a non-preemptive discipline in which waiting job (or process) with It is run smaller amount the completion of the run time to the next. In other words, when a CPU is available for the use, this is assigned to the process that has smallest next CPU burst and the SJF scheduling is especially appropriate for batch jobs for which the run times it well known in anticipation, Where the SJF scheduling algorithm allows the average minimum time for specific a set of processes, it is probably optimal. This algorithm favors short jobs (or processors) at the expense of longer ones. It the best algorithm can rely on user estimates of run times [1].Scheduling is done by examining the length of the next CPU burst of a process. If the CPU is free, the next process with the smallest next CPU burst is assigned. If two processes have the same CPU burst, FIFO is used to break the tie. The difficulty with SJF is to determine the length of the next process. The advantage of this algorithm is that it is optimal by providing the minimum average waiting time and the Shortest-Job-First (SJF) is a non-preemptive discipline in which waiting job (or process) with the smallest estimated Running time to complete, and the next run. In other words, when a CPU is available, it is assigned to the process that has smallest next CPU burst. The shortest job first algorithm favors short jobs (or processors) at the expense of longer ones [1].

Shortest-Job-First Scheduling:

a) Associate with each process the length of its next CPU burst .Use these lengths to schedule the process with the shortest time and the algorithm have two versions or schemes.

Non-preemptive Scheduling:

Once CPU given to the process it cannot be preempted until completes its CPU burst.

Processes release the CPU only after completion. And the Processes releases CPU voluntarily. Examples of Non-preemptive scheduling algorithms: first come first serve (FCFS) or first in first out (FIFO), shortest job first (SJF), and priority.

Preemptive Scheduling:

If a new process arrives with CPU burst length less than remaining time of current executing process, preempt. And this scheme is known as the Shortest-Remaining-Time-First (SRTF). Allows for running processes to be temporarily suspended. Processes releases CPU upon receiving a command.

b) SJF is optimal – gives minimum average waiting time for a given set of processes.

CPU Scheduling:

A CPU scheduler is tasked with choosing which process to run first from the ready queue. A scheduling algorithm is used to choose the next process. Scheduling is a fundamental function of an operating system. A Scheduling refers to picking a process, from many ready-made processes, that is to be next executed on CPU. The CPU scheduling can be defined as the art of determining which processes run on the CPU when there are multiple runnable processes [4,5]. As well as the CPU scheduling decisions may take place when a process:

a) Switches from running to waiting state.

b) Switches from running to ready state.

c) Switching from waiting to ready.

d) Terminates.

Scheduling Criteria:

Different CPU scheduling algorithms have distinct properties, and picking a particular algorithm may favor one class of processes on the other, in choose any algorithm for use in a specific case, we must consider the properties of the various algorithms. Many criteria the different in comparing in the CPU Scheduling algorithms. Which properties which are used for comparison can make a large in the difference which algorithm is judged to be best. The criteria include the following [6,7].

a) **CPU Utilization**: Is the average function of time,, during which the processor is busy.

b) **Through Put**: Refers to the amount of work completed in a unit of time.

c) **Turnaround Time**: Is defined as the time that elapses from the moment a program or a job is submitted until it is completed by a system

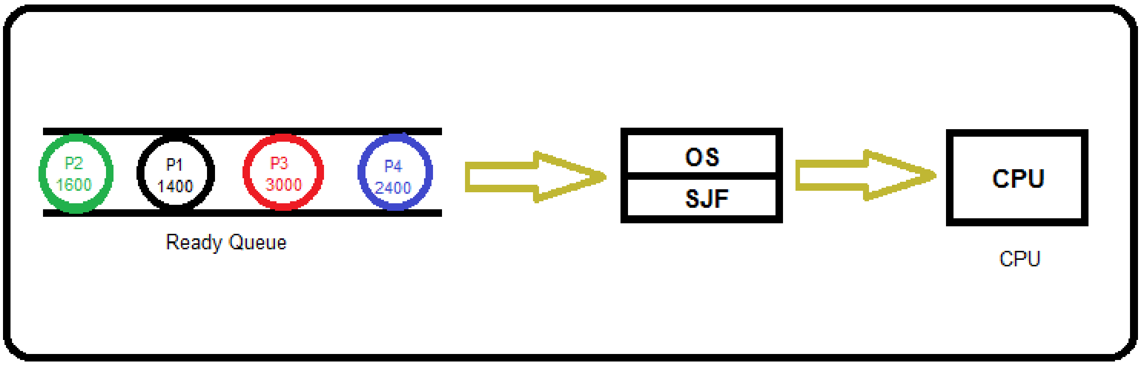
d) **Waiting Time**: The average period of time a process spends waiting.

e) **Response time**: Is a time from the submission of a request until the first response is produced.

f) **Fairness:** To avoid the process from the starvation.

g) **Priority enforcement:** Scheduling policy favors higher priority processes.

And we are explain a block diagram of the shortest job first processing:



Function

All the process wait in to the ready queue for the CPU and the SJF algorithm select the process which has shortest burst time . And showing the program design used to explain the SJF algorithm with code used in visual basic language .

The Program Code Explain

The first explain definition the four dimension and each dimension include 4 value.

The first dimension included 4 Integer value of account processor from (1 to 4).

The second dimension included 4 Integer value of account wait _time (1 to 4).

The third dimension included 4 Integer value of account arrange_ p (1 To 4).

The fourth dimension included 4 Integer value account response (1 To 4).

My program Consists of 4 main dimensions and the Variables are introduced through the input box and the variables are printed by print (p) and is repeated for each matrix of dimensions through for (loop).

The designed program and the implementation consists of three buttons.

The command 1 : for press ⟶ **SJF**

In the first stage we enter the values of the four processors, and as in the example:

```
For i = 1 To 4

p(i) = Input Box("ENTER CPU BURST", "Note")

Print "p"; i

Next                                                                    i
```

CPU 1400 1600 3000 2400

Your processes are :

P1	1400
P2	1600
P3	3000
P4	2400

And after repairing :

P1	1400
P2	1600
P4	2400
P3	3000

And after entered the 4 values we will calculate the total processors and sequence of work Low to High, and we also calculate the time it takes for each processor to perform the operation and after calculating the time per processor for processing times, we calculate the total times to perform the operation (execution of process).

The processors are arranged in terms of time & it takes for processing each processor on another processor are compared with each other and arranged form the bottom to the top in terms of time.

In the first step we enter the values as reported previously and explained in the report and pictures, the program will ask us to enter the four values and stops at the introduction of the fourth value and applies the operations of steps

The command 2 press → **Total wait time**

And account the total time waiting by sum all time for four processes as in example =8400

```
For i = 1 To 4

sum = sum + p(i)

Next i

Print "Total Time of waiting is" & sum .
```

But in The command 3 press → **Response time**

When going through all stages of processing and print output for each step (dimension) and only in the dimension of the response change p1 by p0 and printing because the first value for the processor (CPU) empty(out of work) and the operating system will directly execute the process so the time is = 0 and wait until the intervention of the second operation to the for-loop start from 2 to 4.

We will see for execute

P0	0
P2	1400
P4	3000
P3	5400

```
Print "each process response in sequence"
Print "p" & response(1), 0
For i = 2 To 4
response(i) = Val(response(i - 1)) + Val(p(i - 1))
Print "p" & arrange _p(i), response(i)
Next i
```

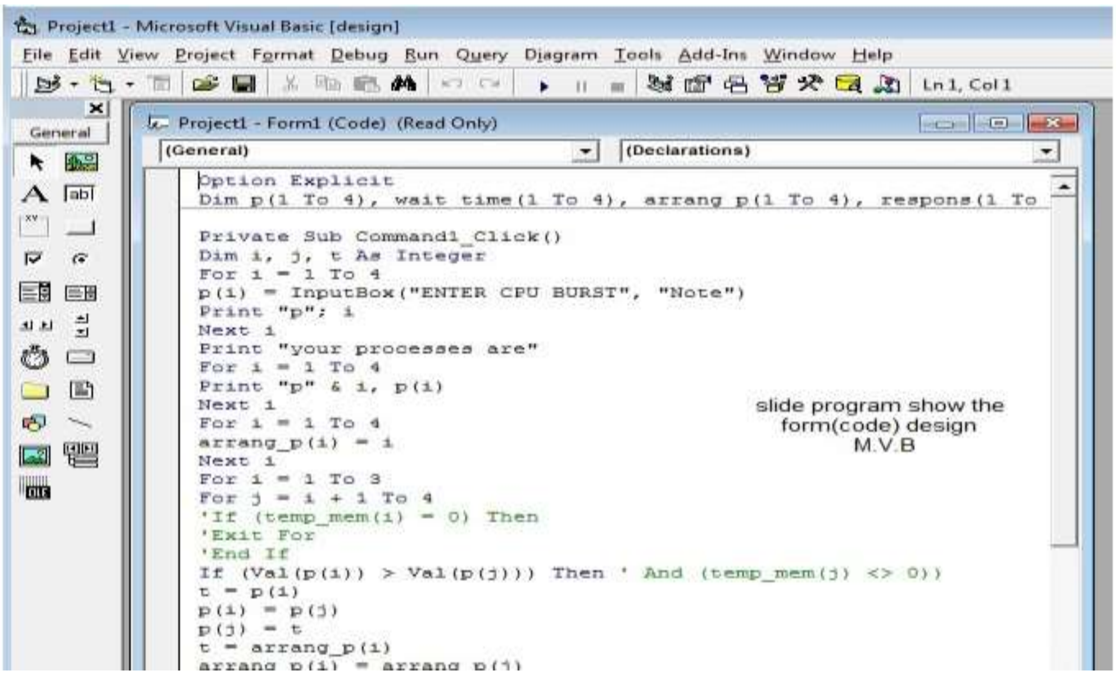
this is the code of program used it with a figures:

```
Option Explicit
Dim p(1 To 4), wait_ time(1 To 4), arrang _ p(1 To 4), respons (1 To 4) As
Private Sub Command1_Click()
Dim i, j, t As Integer
For i = 1 To 4
p(i) = InputBox("ENTER CPU BURST", "Note")
Print "p"; i
Next i
Print "your processes are"
For i = 1 To 4
Print "p" & i, p(i)
Next i
For i = 1 To 4
arrang_p(i) = i
```

```
Next i
For i = 1 To 3
For j = i + 1 To 4
'If (temp_mem(i) = 0) Then
'Exit For
'End If
If (Val(p(i)) > Val(p(j))) Then ' And (temp_mem(j) <> 0))
t = p(i)
p(i) = p(j)
p(j) = t

t = arrang_p(i)
arrang_p(i) = arrang_p(j)
arrang_p(j) = t
End If
Next j
Next i
Print "after repairing"
For i = 1 To 4
Print "p" & arrang_p(i), p(i)
Next i
Print "each process wait in sequence"
Print "p" & arrang_p(1), 0
For i = 2 To 4
wait_time(i) = Val(wait_time(i - 1)) + Val(p(i – 1))
Print "p" & arrang_p(i), wait_time(i)
Next i
End Sub
```

```
Private Sub Command2_Click()
Dim sum, i As Integer
sum = 0
For i = 1 To 4
sum = sum + p(i)
Next i
Print "Total Time of waitnig is" & sum
End Sub
Private Sub Command3_Click()
Dim i As Integer
Print "each process response in sequence"
Print "p" & respons(1), 0
For i = 2 To 4
respons(i) = Val(respons(i - 1)) + Val(p(i – 1))
Print "p" & arrang_p(i), respons(i)
Next i
End Sub
```

And now we will see the first figure for execution the program press in (figure 1).



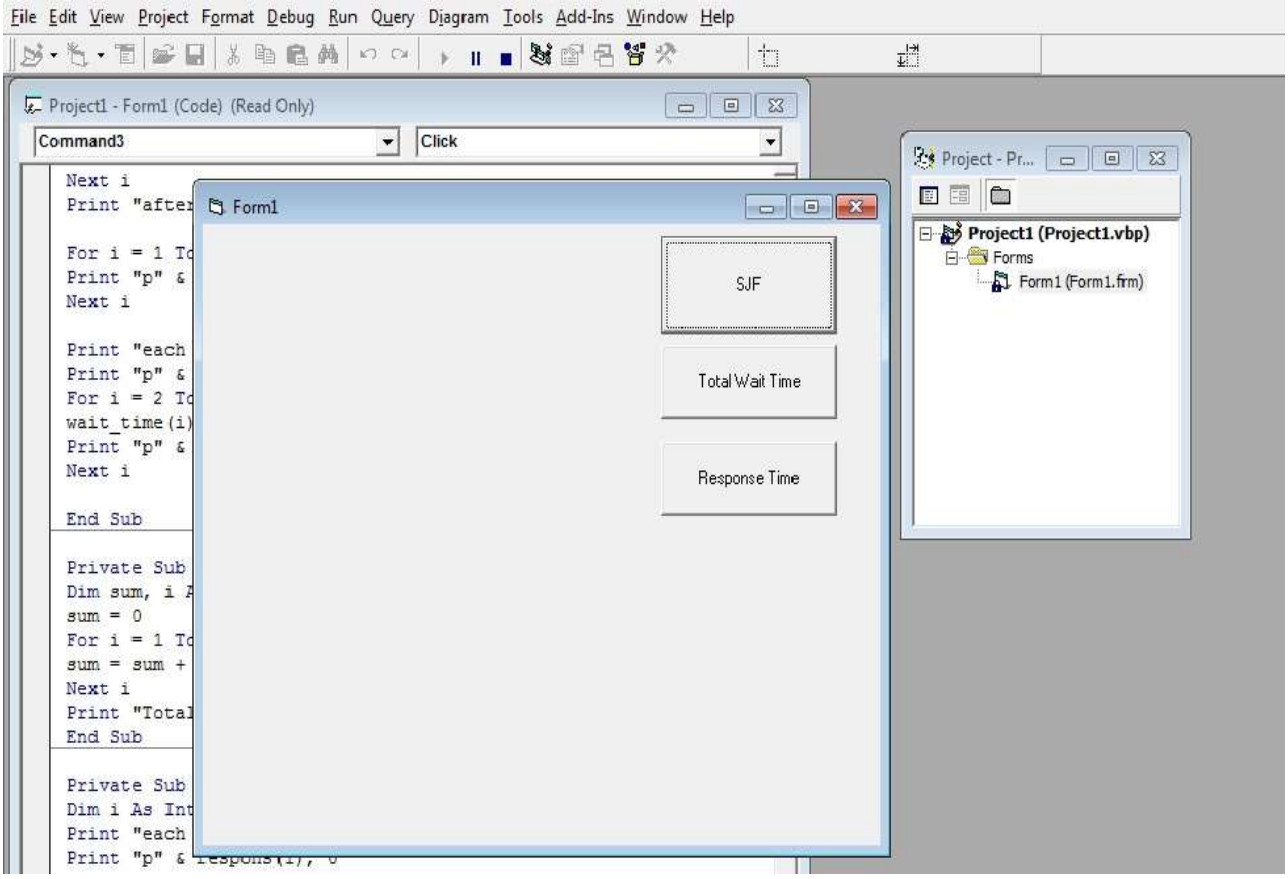
Project1 - Microsoft Visual Basic [design]
File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
Ln 1, Col 1
General
Project1 - Form1 (Code) (Read Only)
(General)
(Declarations)

```
Option Explicit
Dim p(1 To 4), wait_time(1 To 4), arrang_p(1 To 4), respons(1 To

Private Sub Command1_Click()
Dim i, j, t As Integer
For i = 1 To 4
p(i) = InputBox("ENTER CPU BURST", "Note")
Print "p"; i
Next i
Print "your processes are"
For i = 1 To 4
Print "p" & i, p(i)
Next i
For i = 1 To 4
arrang_p(i) = i
Next i
For i = 1 To 3
For j = i + 1 To 4
'If (temp_mem(i) = 0) Then
'Exit For
'End If
If (Val(p(i)) > Val(p(j))) Then ' And (temp_mem(j) <> 0))
t = p(i)
p(i) = p(j)
p(j) = t
t = arrang_p(i)
arrang_p(i) = arrang_p(j)
```

slide program show the
form(code) design
M.V.B

Figure (1)

And enter the 4 value after Press the SJF icon in (figure 2) .

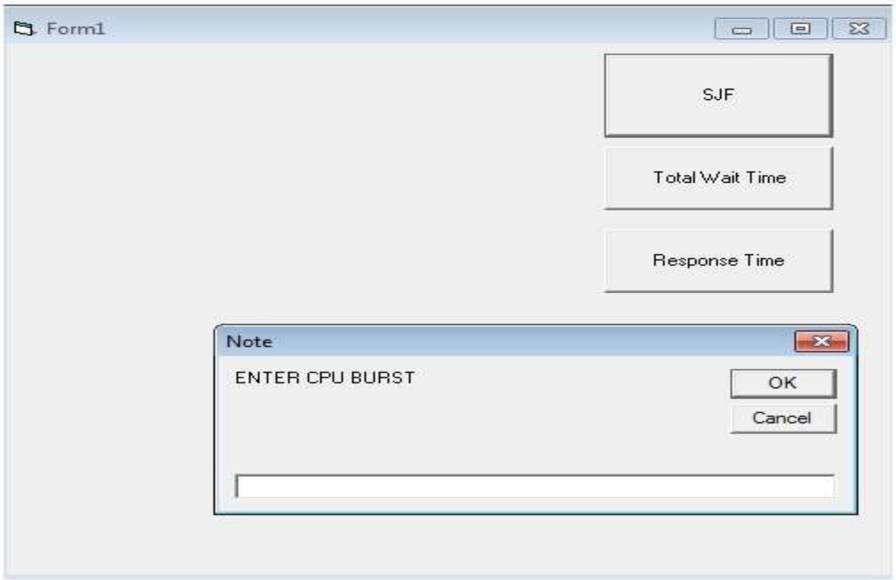


File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
Project1 - Form1 (Code) (Read Only)
Command3
Click
Form1
SJF
Total Wait Time
Response Time
Project - Pr...
Project1 (Project1.vbp)
Forms
Form1 (Form1.frm)

```
Next i
Print "after

For i = 1 To
Print "p" &
Next i

Print "each
Print "p" &
For i = 2 To
wait_time(i)
Print "p" &
Next i

End Sub

Private Sub
Dim sum, i A
sum = 0
For i = 1 To
sum = sum +
Next i
Print "Total
End Sub

Private Sub
Dim i As Int
Print "each
Print "p" & respons(i), 0
```

Figure (2)

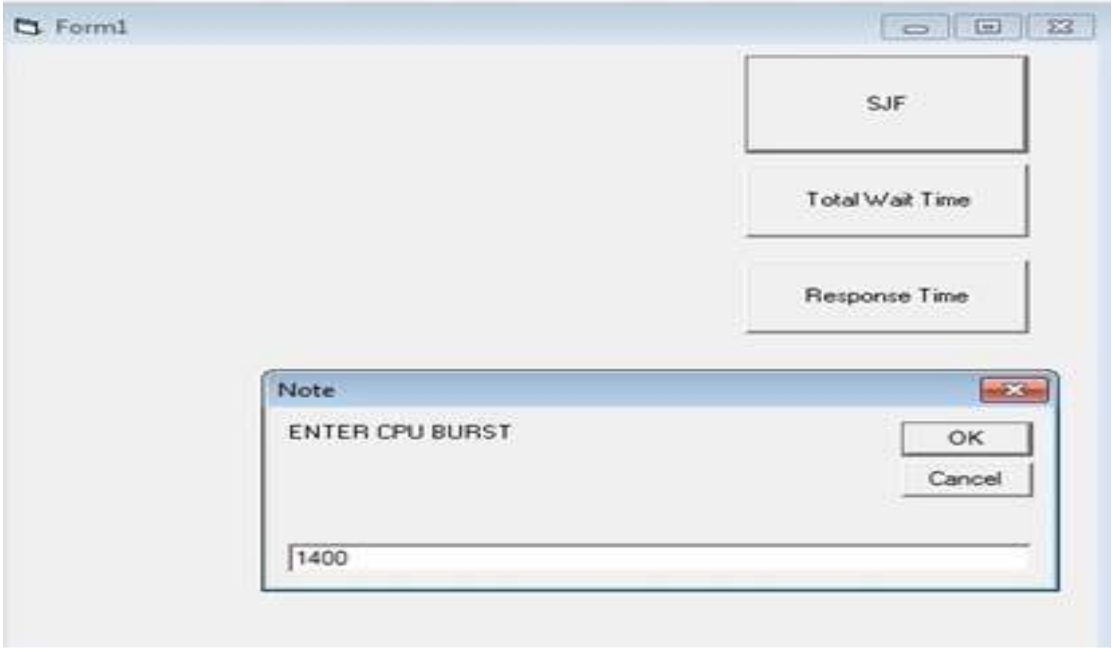
Enter the forth value of CPU burst in (figure 3).



ENTER CPU BURST

Figure (3)

Enter first value of CPU Burst in (figure 4).

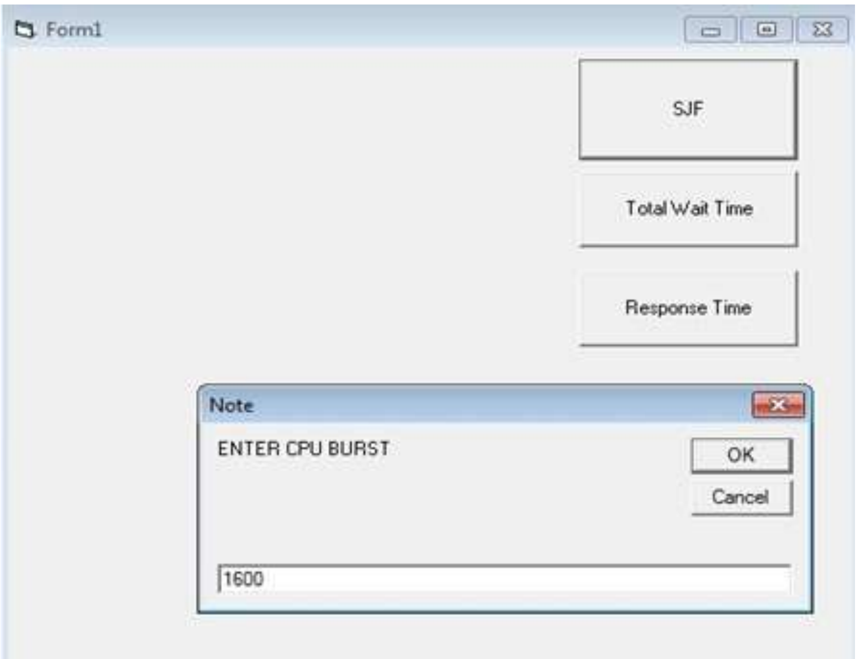


ENTER CPU BURST

1400

Figure (4)

And next the second value of CPU Burst in (figure 5) .

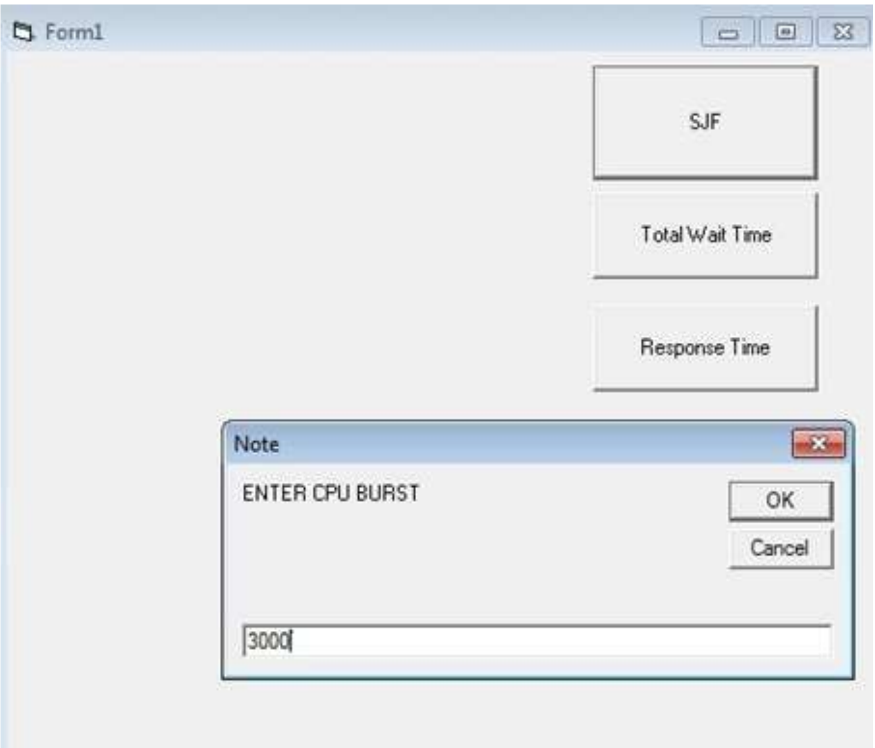


ENTER CPU BURST

1600

Figure (5)

And next the third value of CPU Burst in (figure 6).

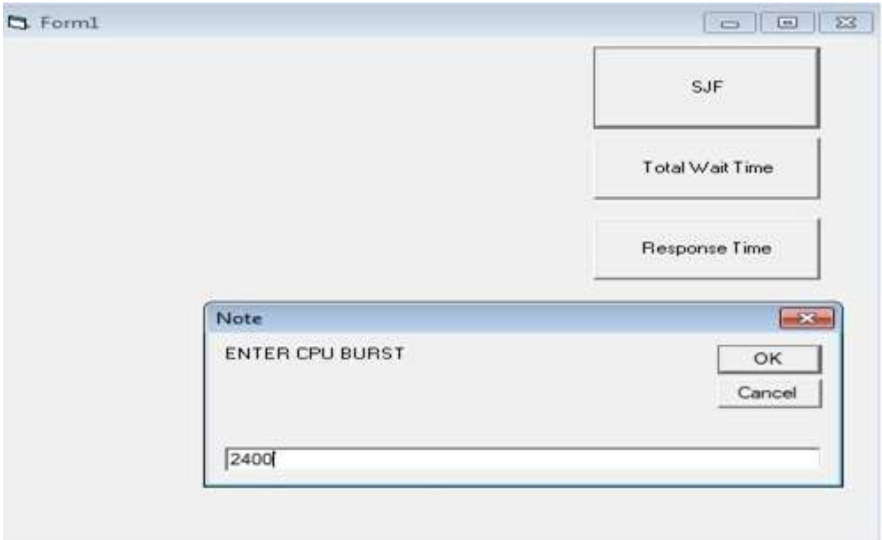


ENTER CPU BURST

3000

Figure (6)

And last the fourth value of CPU Burst in (figure 7).



ENTER CPU BURST

2400

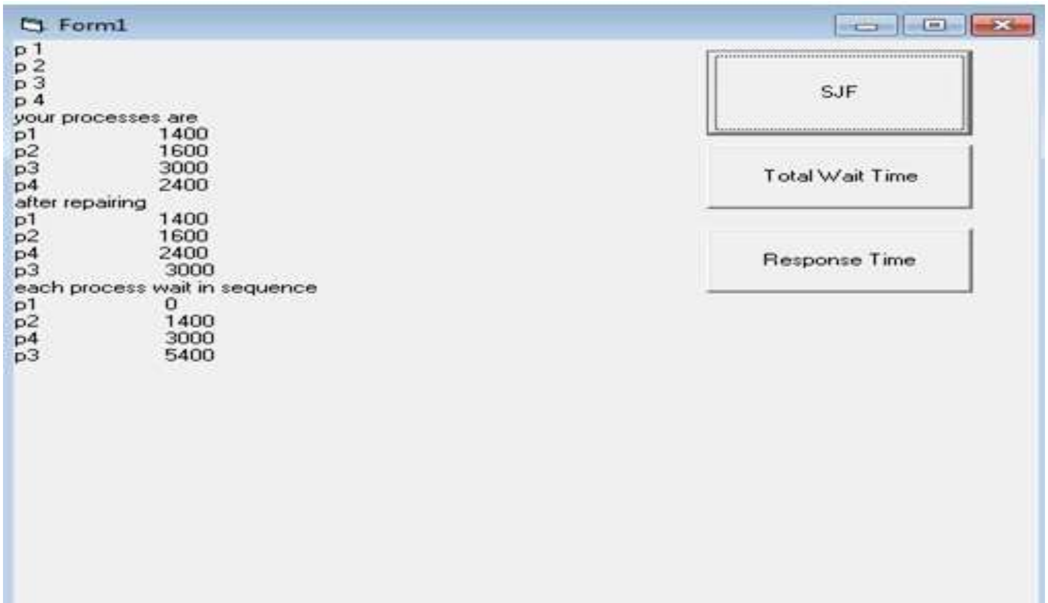
Figure (7)

the result of the entry of all values in the output program screen :

1. The 4 values are entered as p1,p2,p3,p4 in the command .(1400,1600,3000 &2400)

2. The value after repairing is from the slow value to the high value is p1,p2,p4,p3 (1400,1600,2400 and 3000)

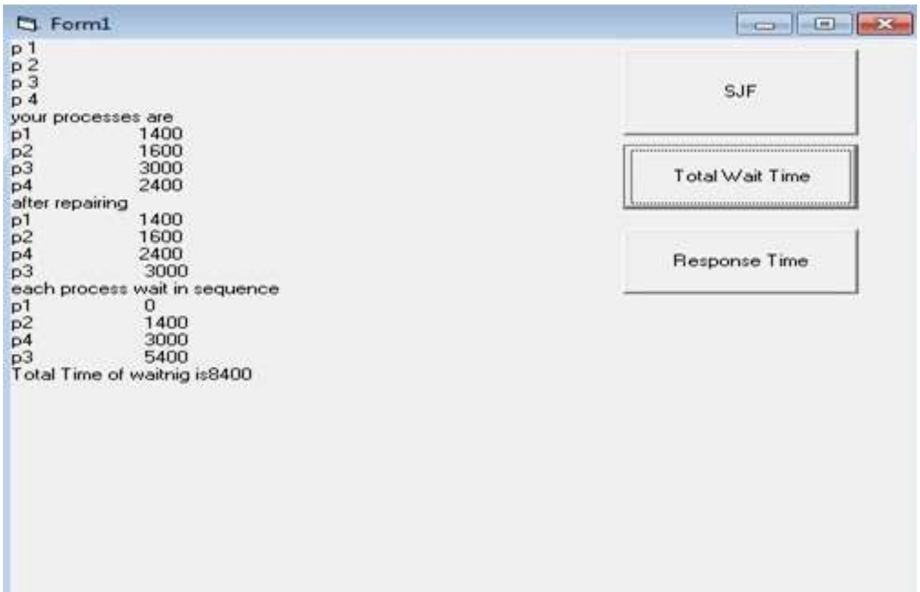
3. And we see the sequence of each process wait from slow to high and se can see print 0 value here the p1 =o for time execute because the CPU of O.S is empty and directly execute the first CPU Burst .sequence from slower to higher explain in (figure 8) .



```
p 1
p 2
p 3
p 4
your processes are
p1           1400
p2           1600
p3           3000
p4           2400
after repairing
p1           1400
p2           1600
p4           2400
p3            3000
each process wait in sequence
p1            0
p2            1400
p4            3000
p3            5400
```

Figure (8)

And now when press the icon of total wait time we will see in (figure 9) .

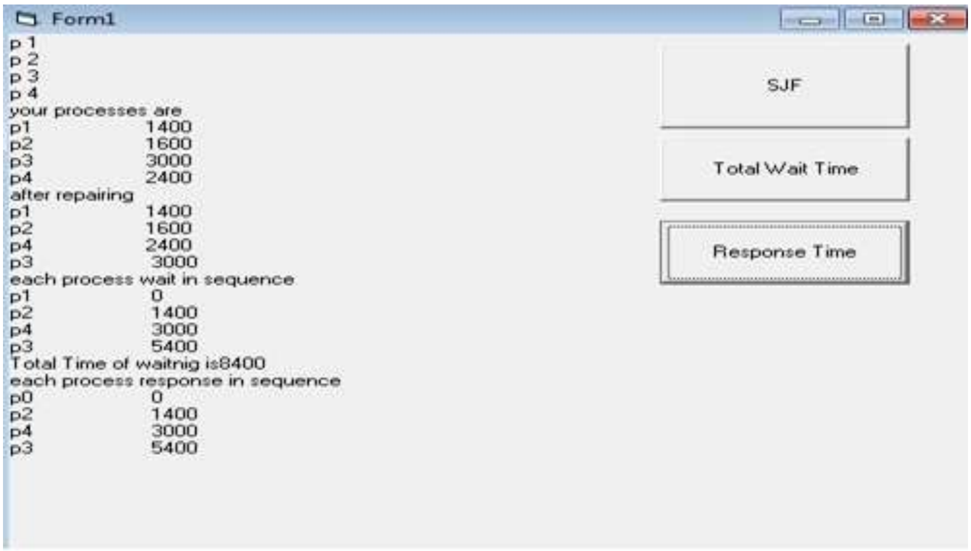


```
p 1
p 2
p 3
p 4
your processes are
p1          1400
p2          1600
p3          3000
p4          2400
after repairing
p1          1400
p2          1600
p4          2400
p3           3000
each process wait in sequence
p1           0
p2           1400
p4           3000
p3           5400
Total Time of waitnig is8400
```

Figure (9)

*Show the print total time of waiting is sum the all wait time of all CPU(processes) is=8400 this result from the sum of each processor (p1+p2+p3+p4) .

And now press the icon of response time we can see in (figure 10) .



```
p 1
p 2
p 3
p 4
your processes are
p1          1400
p2          1600
p3          3000
p4          2400
after repairing
p1          1400
p2          1600
p4          2400
p3           3000
each process wait in sequence
p1           0
p2           1400
p4           3000
p3           5400
Total Time of waitnig is8400
each process response in sequence
p0           0
p2           1400
p4           3000
p3           5400
```

Figure (10)

Here each process response in sequence for work time for execution job of all processors for (wait time for each Pn) p1=0 if the processor of O.S is empty (not work) and execute the first job without any interrupt or problem ,this is no time delay and sequence from the slow --- to ----high CPU(process) work time and work execute, and the wait of p2=1400 execute time of p1,and the wait of p4=3000 time of execute of sum (p1+p2),and for p3= sum the wait time of execute(process) all of (p4+p2+p1) .here p1=p0 print it in program , so the explain of the first job is the CPU of O.S is empty and directly execute the first process (p1) entry and wait is =o here print if enter the p1 in response step is print it in program p0 ----- 0 .

CONCLUSION

In this paper, the shortest job first algorithm is recommended the CPU scheduling problems of minimizing either the average waiting time or average turnaround time, and also the SJF scheduling algorithm is preemptive and non-preemptive. Assume that there is a minimum schedule that is not Shortest Job First and has a lower average waiting time than it explained from the (CPU Burst1 to CPU Burst4) When the executing the application program by used the SJF algorithm after input of the first CPU Burst, according to the shortest time necessary to complete it the last CPU Burst and calculated the total wait time to executed all processes and finally we founding the response time between all processes . As well as scheduling SJF algorithm is often used in long-term scheduling.

REFERENCES

[1] Abraham Silberschatz, Greg Gagne, and Peter Baer Galvin, "Operating System Concepts, Ninth Edition ", Chapter 6.
[2] Nikos P . Ouzounidis, "Performance Evaluation of the SJF Scheduling Algorithm in a Multiprocessing Environment";Florida Institute of Technology;1989 .
[3] Brian Siler and Jeff Spotts; "Using Visual Basic 6.0, Special Edition"; Que publishing; 1998.
[4] Englander, I., 2003. The Architecture of Computer Hardware and Systems Software; An Information Technology Approach, 3rd Edition, John Wiley & Sons, Inc.,
[5] H.H.S. Lee; Lecture: CPU Scheduling, School of Electrical and Computer Engineering, Georgia Institute of Technology.
[6] Silberschatz. A, Galvin. P.B. Gagne. G" Operating system principles,7th Edition " .
[7] I.A. Dhotre "Operating Systems, Eighth Revised Edition;2009 " .