**CPU SCEDULING CODE**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace scheduler\_CPU

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

**#region FCFS Code mode**

**/\* ------------------------ FCFS ----------------------------------- \*/**

int n\_fcfs;//number of processes

private void button1\_Click(object sender, EventArgs e)

{

n\_fcfs = int.Parse(nOfProcesses.Text);

MessageBox.Show("ok!number of processes inserted");

flowLayoutPanel\_fcfs.Controls.Clear();

flowLayoutPanel\_fcfs\_nums.Controls.Clear();

button2.Enabled=true;

counter\_fcfs = 0;

averageWtime\_fcfs = 0;

time\_fcfs = 0;

}

Queue<Process> queue\_fcfs = new Queue<Process>();

int counter\_fcfs = 0;

float averageWtime\_fcfs = 0;

private void button2\_Click(object sender, EventArgs e)

{

Process p\_fcfs = new Process(); p\_fcfs.Pid = counter\_fcfs + 1;

p\_fcfs.burst\_time = int.Parse(burstText\_fcfs.Text);

p\_fcfs.arrival\_time = int.Parse(arrivalText\_fcfs.Text);

queue\_fcfs.Enqueue(p\_fcfs);

counter\_fcfs++;

string ss\_fcfs = "Inserted " + (counter\_fcfs.ToString()) + " from " + (n\_fcfs.ToString());

MessageBox.Show(ss\_fcfs);

if (n\_fcfs == counter\_fcfs)

{

button2.Enabled = false;

calculateWaitingTime\_fcfs();

}

}

int time\_fcfs = 0;

private void calculateWaitingTime\_fcfs()

{

Process tempp\_fcfs;

while (queue\_fcfs.Count != 0)

{

tempp\_fcfs = queue\_fcfs.Dequeue();

if (tempp\_fcfs.arrival\_time > time\_fcfs)

{

queue\_fcfs.Enqueue(tempp\_fcfs);

}

else

{

tempp\_fcfs.waiting\_time = time\_fcfs - tempp\_fcfs.arrival\_time;

averageWtime\_fcfs += tempp\_fcfs.waiting\_time;

//time chart//

Label num\_fcfs = new Label();

num\_fcfs.Text = time\_fcfs.ToString();

if (tempp\_fcfs.burst\_time < 3) num\_fcfs.Width = 27;

else num\_fcfs.Width = tempp\_fcfs.burst\_time \* 10;

flowLayoutPanel\_fcfs\_nums.Controls.Add(num\_fcfs);

//end time chart//

time\_fcfs += tempp\_fcfs.burst\_time;

//gantt chart code

Label tempLabel\_fcfs = new Label();

tempLabel\_fcfs.Enabled = true;

tempLabel\_fcfs.BorderStyle = BorderStyle.FixedSingle;

tempLabel\_fcfs.Font = new Font("Arial", 10, FontStyle.Bold);

string s = "P" + tempp\_fcfs.Pid.ToString();

tempLabel\_fcfs.Text = s;

if (tempp\_fcfs.burst\_time < 3) tempLabel\_fcfs.Width = 27;

else tempLabel\_fcfs.Width = tempp\_fcfs.burst\_time \* 10;

flowLayoutPanel\_fcfs.Controls.Add(tempLabel\_fcfs);

////////////////////////

tempp\_fcfs.burst\_time -= tempp\_fcfs.burst\_time;

tempp\_fcfs.last\_active = time\_fcfs;

}

}

//last time label

Label num\_fcfsLast = new Label();

num\_fcfsLast.Text = time\_fcfs.ToString();

flowLayoutPanel\_fcfs\_nums.Controls.Add(num\_fcfsLast);

//////////////////

waitingText\_fcfs.Text = (averageWtime\_fcfs / n\_fcfs).ToString();

}

**/\* -------------------------- END FCFS ------------------------------ \*/**

**#endregion**

**#region SJF mode Code**

**/\*---------------------- SJF mode Code --------------------------- \*/**

// number of process

int num\_process;

private void numericUpDown1\_ValueChanged(object sender, EventArgs e)

{

num\_process = (int)numericUpDown1.Value;

// reset and start again

button3.Enabled = true;

// gant.Clear();

flowLayoutPanel5.Controls.Clear();

flowLayoutPanel4.Controls.Clear();

sjwtime = 0;

sjaverageWtime = 0;

prsjaverageWtime = 0;

prsjwtime = 0;

pd = 1;

}

// Preemptive check box

private void checkBox1\_Checked(object sender, EventArgs e)

{

if (checkBox1.Checked)

{

label7.Visible = true;

numericUpDown3.Visible = true;

}

else

{

label7.Visible = false;

numericUpDown3.Visible = false;

}

}

// store values accending

double sjwtime = 0, sjaverageWtime = 0, prsjaverageWtime = 0, prsjwtime = 0;

List<Process> vals = new List<Process>();

List<Process> premtive\_vals = new List<Process>();

int pd = 1;

private void button3\_Click(object sender, EventArgs e)

{

Process x = new Process(); // dummy process

x.Pid = pd;

pd++;

x.burst\_time = (int)numericUpDown2.Value;

if (checkBox1.Checked) x.arrival\_time = (int)numericUpDown3.Value;

vals.Add(x);

string ss\_sjf = "Inserted " + (vals.Count.ToString()) + " from " + (num\_process.ToString());

MessageBox.Show(ss\_sjf);

if (vals.Count == num\_process && !checkBox1.Checked)

{

button3.Enabled = false;

vals = vals.OrderBy(arr => arr.burst\_time).ToList();

int st\_t = 0;

for (int j = 0; j < num\_process; j++)

{

//gantt chart code

///////// type time lable

Label num\_fcfs = new Label();

num\_fcfs.Location = new Point(10 + st\_t \* 10, 305);

num\_fcfs.Text = st\_t.ToString();

if (vals[j].burst\_time < 3) num\_fcfs.Width = 28;

else num\_fcfs.Width = vals[j].burst\_time \* 10 - 2;

flowLayoutPanel5.Controls.Add(num\_fcfs);

/////////

st\_t += vals[j].burst\_time;

//// type process number lable

Label tempLabel\_fcfs = new Label();

tempLabel\_fcfs.Enabled = true;

tempLabel\_fcfs.BorderStyle = BorderStyle.FixedSingle;

tempLabel\_fcfs.Font = new Font("Arial", 10, FontStyle.Bold);

tempLabel\_fcfs.Location = new Point(10 + time\_fcfs \* 10, 300);

string s = "P" + vals[j].Pid.ToString();

tempLabel\_fcfs.Text = s;

tempLabel\_fcfs.TextAlign = ContentAlignment.MiddleCenter;

if (vals[j].burst\_time < 3) tempLabel\_fcfs.Width = 30;

else tempLabel\_fcfs.Width = vals[j].burst\_time \* 10;

flowLayoutPanel4.Controls.Add(tempLabel\_fcfs);

////////////////////////

//cal average waiting time

sjaverageWtime += sjwtime;

sjwtime += vals[j].burst\_time;

}

//print last timelable in gant

Label num\_fcfss = new Label();

num\_fcfss.Location = new Point(10 + st\_t \* 10, 300);

num\_fcfss.Text = st\_t.ToString();

flowLayoutPanel5.Controls.Add(num\_fcfss);

/////////////////

//print average time

label9.Text = (sjaverageWtime / (double)num\_process).ToString() + " msec";

vals.Clear();

}

//preemtive code

else if (vals.Count == num\_process && checkBox1.Checked)

{

button3.Enabled = false;

vals = vals.OrderBy(arr => arr.arrival\_time).ToList();

int indx = 0;

for (int i = 0; i < vals.Count - 1; i++)

{

Process pre\_sjf = new Process();

pre\_sjf.burst\_time = vals[i + 1].arrival\_time - vals[i].arrival\_time;

pre\_sjf.Pid = vals[indx].Pid;

pre\_sjf.arrival\_time = vals[indx].arrival\_time;

premtive\_vals.Add(pre\_sjf);

vals[indx].burst\_time -= pre\_sjf.burst\_time;

int bst\_tm = vals[indx].burst\_time;

for (int j = i + 1; j >= 0; j--)

if (vals[j].burst\_time < bst\_tm) indx = j;

}

vals = vals.OrderBy(arr => arr.burst\_time).ToList();

for (int xx = 0; xx < vals.Count; xx++) premtive\_vals.Add(vals.ElementAt(xx));//join lists

for (int jj = 0; jj < premtive\_vals.Count - 1; jj++)//join similar PID

{

if (premtive\_vals[jj].Pid == premtive\_vals[jj + 1].Pid)

{

premtive\_vals[jj + 1].burst\_time += premtive\_vals[jj].burst\_time;

premtive\_vals.RemoveAt(jj);

jj = -1;

}

}

int st\_t = 0; int bs = 0;

for (int j = 0; j < premtive\_vals.Count; j++)

{

//gantt chart code

///////// type time lable

Label num\_fcfs = new Label();

num\_fcfs.Location = new Point(10 + st\_t \* 10, 305);

num\_fcfs.Text = st\_t.ToString();

if (premtive\_vals[j].burst\_time < 3) num\_fcfs.Width = 28;

else num\_fcfs.Width = premtive\_vals[j].burst\_time \* 10 - 2;

flowLayoutPanel5.Controls.Add(num\_fcfs);

/////////

st\_t += premtive\_vals[j].burst\_time;

//// type process number lable

Label tempLabel\_fcfs = new Label();

tempLabel\_fcfs.Enabled = true;

tempLabel\_fcfs.BorderStyle = BorderStyle.FixedSingle;

tempLabel\_fcfs.Font = new Font("Arial", 10, FontStyle.Bold);

tempLabel\_fcfs.Location = new Point(10 + time\_fcfs \* 10, 300);

string s = "P" + premtive\_vals[j].Pid.ToString();

tempLabel\_fcfs.Text = s;

tempLabel\_fcfs.TextAlign = ContentAlignment.MiddleCenter;

if (premtive\_vals[j].burst\_time < 3) tempLabel\_fcfs.Width = 30;

else tempLabel\_fcfs.Width = premtive\_vals[j].burst\_time \* 10;

flowLayoutPanel4.Controls.Add(tempLabel\_fcfs);

////////////////////////

//cal average waiting time

for (int z = j - 1; z >= 0; z--)

if (premtive\_vals[j].Pid == premtive\_vals[z].Pid)

bs = premtive\_vals[z].burst\_time;

prsjaverageWtime += (prsjwtime - bs - premtive\_vals[j].arrival\_time);

bs = 0;

prsjwtime += premtive\_vals[j].burst\_time;

}

//print last timelable in gant

Label num\_fcfss = new Label();

num\_fcfss.Location = new Point(10 + st\_t \* 10, 300);

num\_fcfss.Text = st\_t.ToString();

flowLayoutPanel5.Controls.Add(num\_fcfss);

/////////////////

//print avg

label9.Text = (prsjaverageWtime / (double)num\_process).ToString() + " msec";

vals.Clear();

premtive\_vals.Clear();

}

}

**/\*---------------------------- End SJF mode Code-----------------------------\*/**

**#endregion**

**#region priority mode code**

**/\*----------------------------- priority mode code---------------------------\*/**

//num of process

int num\_process\_prio;

private void numericUpDown5\_ValueChanged(object sender, EventArgs e)

{

num\_process\_prio = (int)numericUpDown5.Value;

//reset values and start again

button4.Enabled = true;

flowLayoutPanel7.Controls.Clear();

flowLayoutPanel8.Controls.Clear();

prwtime = 0;

praverageWtime = 0;

prtpraverageWtime = 0;

prtprwtime = 0;

pr\_pd = 1;

}

// Preemptive checkbox

private void checkBox2\_CheckedChanged(object sender, EventArgs e)

{

if (checkBox2.Checked)

{

label14.Visible = true;

numericUpDown6.Visible = true;

}

else

{

label14.Visible = false;

numericUpDown6.Visible = false;

}

}

// store values and priority

double prwtime = 0, praverageWtime = 0, prtprwtime = 0, prtpraverageWtime = 0; int pr\_pd = 1;

List<Process> pri\_vals = new List<Process>();

List<Process> premt\_pri\_vals = new List<Process>();

private void button4\_Click(object sender, EventArgs e)

{

Process prix = new Process();

int[] process = new int[2];

prix.Pid = pr\_pd;

pr\_pd++;

prix.burst\_time = (int)numericUpDown4.Value; //time

prix.priority = (int)numericUpDown7.Value; //priority

if (checkBox2.Checked) prix.arrival\_time = (int)numericUpDown6.Value; //arival time

pri\_vals.Add(prix);

string ss\_prio = "Inserted " + (pri\_vals.Count.ToString()) + " from " +( num\_process\_prio.ToString());

MessageBox.Show(ss\_prio);

if (pri\_vals.Count == num\_process\_prio && !checkBox2.Checked)

{

button4.Enabled = false;

pri\_vals = pri\_vals.OrderBy(arr => arr.priority).ToList(); //sorting based on priority

int st\_t = 0;

for (int j = 0; j < num\_process\_prio; j++)

{

//gantt chart code

///////// type time lable

Label num\_fcfs = new Label();

num\_fcfs.Location = new Point(10 + st\_t \* 10, 305);

num\_fcfs.Text = st\_t.ToString();

if (pri\_vals[j].burst\_time < 3) num\_fcfs.Width = 28;

else num\_fcfs.Width = pri\_vals[j].burst\_time \* 10 - 2;

flowLayoutPanel8.Controls.Add(num\_fcfs);

/////////

st\_t += pri\_vals[j].burst\_time;

//// type process number lable

Label tempLabel\_fcfs = new Label();

tempLabel\_fcfs.Enabled = true;

tempLabel\_fcfs.BorderStyle = BorderStyle.FixedSingle;

tempLabel\_fcfs.Font = new Font("Arial", 10, FontStyle.Bold);

tempLabel\_fcfs.Location = new Point(10 + time\_fcfs \* 10, 300);

string s = "P" + pri\_vals[j].Pid.ToString();

tempLabel\_fcfs.Text = s;

tempLabel\_fcfs.TextAlign = ContentAlignment.MiddleCenter;

if (pri\_vals[j].burst\_time < 3) tempLabel\_fcfs.Width = 30;

else tempLabel\_fcfs.Width = pri\_vals[j].burst\_time \* 10;

flowLayoutPanel7.Controls.Add(tempLabel\_fcfs);

////////////////////////

//cal average waiting time

praverageWtime += prwtime;

prwtime += pri\_vals[j].burst\_time;

}

//print last timelable in gant

Label num\_fcfss = new Label();

num\_fcfss.Location = new Point(10 + st\_t \* 10, 300);

num\_fcfss.Text = st\_t.ToString();

flowLayoutPanel8.Controls.Add(num\_fcfss);

/////////////////

/// print average waiting time

label12.Text = (praverageWtime / (double)num\_process\_prio).ToString() + " msec";

pri\_vals.Clear();

}

//preemtive code

else if (pri\_vals.Count == num\_process\_prio && checkBox2.Checked)

{

button4.Enabled = false;

pri\_vals = pri\_vals.OrderBy(arr => arr.arrival\_time).ToList();

int indx = 0;

for (int i = 0; i < pri\_vals.Count - 1; i++)

{

Process pre\_sjf = new Process();

pre\_sjf.burst\_time = pri\_vals[i + 1].arrival\_time - pri\_vals[i].arrival\_time;

pre\_sjf.Pid = pri\_vals[indx].Pid;

pre\_sjf.arrival\_time = pri\_vals[indx].arrival\_time;

premt\_pri\_vals.Add(pre\_sjf);

pri\_vals[indx].burst\_time -= pre\_sjf.burst\_time;

for (int j = i + 1; j >= 0; j--)

if (pri\_vals[j].priority < pri\_vals[indx].priority) indx = j;

}

pri\_vals = pri\_vals.OrderBy(arr => arr.priority).ToList();

for (int xx = 0; xx < pri\_vals.Count; xx++) premt\_pri\_vals.Add(pri\_vals.ElementAt(xx));//join lists

for (int jj = 0; jj < premt\_pri\_vals.Count - 1; jj++)//join similar PID

{

if (premt\_pri\_vals[jj].Pid == premt\_pri\_vals[jj + 1].Pid)

{

premt\_pri\_vals[jj + 1].burst\_time += premt\_pri\_vals[jj].burst\_time;

premt\_pri\_vals.RemoveAt(jj);

jj = -1;

}

}

int st\_t = 0; int bzs = 0;

for (int j = 0; j < premt\_pri\_vals.Count; j++)

{

//gantt chart code

///////// type time lable

Label num\_fcfs = new Label();

num\_fcfs.Location = new Point(10 + st\_t \* 10, 305);

num\_fcfs.Text = st\_t.ToString();

if (premt\_pri\_vals[j].burst\_time < 3) num\_fcfs.Width = 28;

num\_fcfs.Width = premt\_pri\_vals[j].burst\_time \* 10 - 2;

flowLayoutPanel8.Controls.Add(num\_fcfs);

/////////

st\_t += premt\_pri\_vals[j].burst\_time;

//// type process number lable

Label tempLabel\_fcfs = new Label();

tempLabel\_fcfs.Enabled = true;

tempLabel\_fcfs.BorderStyle = BorderStyle.FixedSingle;

tempLabel\_fcfs.Font = new Font("Arial", 10, FontStyle.Bold);

tempLabel\_fcfs.Location = new Point(10 + time\_fcfs \* 10, 300);

string s = "P" + premt\_pri\_vals[j].Pid.ToString();

tempLabel\_fcfs.Text = s;

tempLabel\_fcfs.TextAlign = ContentAlignment.MiddleCenter;

if (premt\_pri\_vals[j].burst\_time < 3) tempLabel\_fcfs.Width = 30;

else tempLabel\_fcfs.Width = premt\_pri\_vals[j].burst\_time \* 10;

flowLayoutPanel7.Controls.Add(tempLabel\_fcfs);

////////////////////////

//cal average waiting time

for (int z = j - 1; z >= 0; z--)

if (premt\_pri\_vals[j].Pid == premt\_pri\_vals[z].Pid)

bzs = premt\_pri\_vals[z].burst\_time;

prtpraverageWtime += (prtprwtime - bzs - premt\_pri\_vals[j].arrival\_time);

bzs = 0;

prtprwtime += premt\_pri\_vals[j].burst\_time;

}

//print last timelable in gant

Label num\_fcfss = new Label();

num\_fcfss.Location = new Point(10 + st\_t \* 10, 300);

num\_fcfss.Text = st\_t.ToString();

flowLayoutPanel8.Controls.Add(num\_fcfss);

/////////////////

/// print average waiting time

label12.Text = (prtpraverageWtime / (double)num\_process\_prio).ToString() + " msec";

pri\_vals.Clear();

premt\_pri\_vals.Clear();

}

}

**/\*-------------------- End priority mode code -------------------------- \*/**

**#endregion**

**#region Round Robin mode code**

**/\*---------------------Round Robin mode code ---------------------------\*/**

int n\_RR;//number of processes

int quantum;

private void button6\_Click\_1(object sender, EventArgs e)

{

n\_RR = int.Parse(textBox2.Text);

quantum = int.Parse(textBox4.Text);

MessageBox.Show("ok!number of processes and quantum time inserted");

flowLayoutPanel\_RR.Controls.Clear();

flowLayoutPanel\_RR\_nums.Controls.Clear();

insertButton\_RR.Enabled = true;

counter\_RR = 0;

averageWtime\_RR = 0;

time\_RR = 0;

}

Queue<Process> queue = new Queue<Process>();

int counter\_RR = 0;

float averageWtime\_RR = 0;

private void insertButton\_RR\_Click(object sender, EventArgs e)

{

Process p = new Process(); p.Pid = counter\_RR + 1;

p.burst\_time = int.Parse(burstText\_RR.Text);

p.arrival\_time = int.Parse(arrivalText\_RR.Text);

queue.Enqueue(p);

counter\_RR++;

string ss\_RR = "Inserted " + (counter\_RR.ToString()) + " from " + (n\_RR.ToString());

MessageBox.Show(ss\_RR);

if (n\_RR == counter\_RR)

{

insertButton\_RR.Enabled = false;

calculateWaitingTime\_RR();

}

}

int time\_RR = 0;

public void calculateWaitingTime\_RR()

{

Process tempp;

while (queue.Count != 0)

{

tempp = queue.Dequeue();

if (tempp.arrival\_time > time\_RR)

{

queue.Enqueue(tempp);

}

else

{

if (tempp.burst\_time < quantum)

{

if(tempp.last\_active==0)

tempp.waiting\_time = time\_RR - tempp.arrival\_time;

else

tempp.waiting\_time = time\_RR - tempp.last\_active;

averageWtime\_RR += tempp.waiting\_time;

//////////

Label num\_RR = new Label();

num\_RR.Text = time\_RR.ToString();

if (tempp.burst\_time < 3) num\_RR.Width = 27;

else num\_RR.Width = tempp.burst\_time \* 10;

flowLayoutPanel\_RR\_nums.Controls.Add(num\_RR);

/////////

time\_RR += tempp.burst\_time;

//gantt chart code

Label tempLabel\_RR = new Label();

tempLabel\_RR.Enabled = true;

tempLabel\_RR.BorderStyle = BorderStyle.FixedSingle;

tempLabel\_RR.Font = new Font("Arial", 10, FontStyle.Bold);

string s = "p" + tempp.Pid.ToString();

tempLabel\_RR.Text = s;

if (tempp.burst\_time < 3) tempLabel\_RR.Width = 27;

else tempLabel\_RR.Width = tempp.burst\_time \* 10;

flowLayoutPanel\_RR.Controls.Add(tempLabel\_RR);

////////////////////////

tempp.burst\_time -= tempp.burst\_time;

tempp.last\_active = time\_RR;

}

else

{

if (tempp.last\_active == 0)

tempp.waiting\_time = time\_RR - tempp.arrival\_time;

else

tempp.waiting\_time = time\_RR - tempp.last\_active;

averageWtime\_RR += tempp.waiting\_time;

//////////

Label num\_RR = new Label();

num\_RR.Text = time\_RR.ToString();

if (quantum < 3) num\_RR.Width = 27;

else num\_RR.Width = quantum \* 10;

flowLayoutPanel\_RR\_nums.Controls.Add(num\_RR);

/////////

time\_RR += quantum;

//gantt chart code

Label tempLabel\_RR = new Label();

tempLabel\_RR.Enabled = true;

tempLabel\_RR.BorderStyle = BorderStyle.FixedSingle;

tempLabel\_RR.Font = new Font("Arial", 10, FontStyle.Bold);

string s = "p" + tempp.Pid.ToString();

tempLabel\_RR.Text = s;

if (quantum < 3) tempLabel\_RR.Width = 27;

else tempLabel\_RR.Width = quantum \* 10;

flowLayoutPanel\_RR.Controls.Add(tempLabel\_RR);

////////////////////////

tempp.burst\_time -= quantum;

tempp.last\_active = time\_RR;

if (tempp.burst\_time != 0) queue.Enqueue(tempp);//20 16

}

//gantt chart code

}

}

//last time label

Label num\_RRLast = new Label();

num\_RRLast.Text = time\_RR.ToString();

flowLayoutPanel\_RR\_nums.Controls.Add(num\_RRLast);

//////////////////

waitingText\_RR.Text = (averageWtime\_RR / n\_RR).ToString();

}

private void tabPage2\_Click(object sender, EventArgs e)

{

}

private void flowLayoutPanel3\_Paint(object sender, PaintEventArgs e)

{

}

**/\*---------------------End Round Robin mode code-------------------------\*/**

**#endregion**

}

}

class Process

{

public int Pid;

public int burst\_time;

public int waiting\_time = 0;

public int last\_active = 0;

public int arrival\_time;

public int priority;

}