Codes and Outputs:

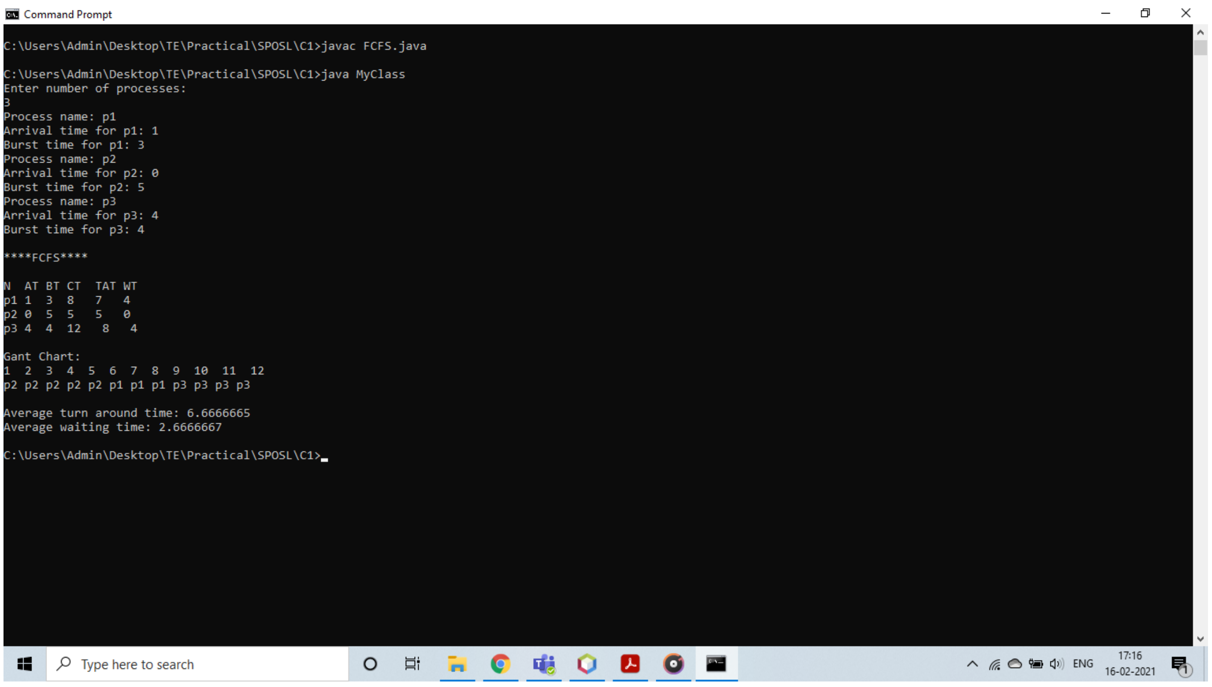
FCFS:

```
import java.util.*;

import java.lang.*;

import java.io.*;


class Process

{

	String name;

	int arrival_time, burst_time;

	Process(String n, int at, int bt)

	{

		name = n;

		arrival_time = at;

		burst_time = bt;

	}

}


class SortByArrivalTime implements Comparator<Process>

{

	public int compare(Process a, Process b)

	{

		return a.arrival_time - b.arrival_time;

	}

}


class SortByBurstTime implements Comparator<Process>

{
	public int compare(Process a, Process b)
	{
		return a.burst_time - b.burst_time;
	}
}

class MyClass
{
	public static void main(String[] args) throws ClassCastException
	{
		Scanner sc = new Scanner(System.in);
		System.out.println("Enter number of processes: ");
		int n = sc.nextInt();
		Vector<Process> process = new Vector<Process>();
		Vector<Process> process_clone = new Vector<Process>();
		Vector<Integer> completion_time = new Vector<Integer>();
		Vector<Integer> turn_around_time = new Vector<Integer>();
		Vector<Integer> waiting_time = new Vector<Integer>();
		Vector<String> gant_chart = new Vector<String>();
		int bt, at;
		String name;
		for(int i=0; i<n; i++)
		{
			sc.nextLine();
			System.out.print("Process name: ");
			name = sc.nextLine();
			System.out.print("Arrival time for " + name + ": ");
			at = sc.nextInt();
			System.out.print("Burst time for " + name + ": ");
```

```
                bt = sc.nextInt();
                process.add(new Process(name, at, bt));
                process_clone.add(new Process(name, at, bt));
            }
            Collections.sort(process, new SortByArrivalTime());
            int flag = 0;
            for(int i=0; i<n; i++)
            {
                if(process.get(i).arrival_time != 0 && flag == 0)
                {
                    for(int j=0; j<process.get(i).arrival_time; j++)
                    {
                        gant_chart.add("n");
                    }
                }
                flag = 1;
                for(int j=0; j<process.get(i).burst_time; j++)
                {
                    gant_chart.add(process.get(i).name);
                }
            }
            for(int i=0; i<n; i++)
            {

        completion_time.add(gant_chart.lastIndexOf(process_clone.get(i).name) + 1);
            }
            for(int i=0; i<n; i++)
            {
                turn_around_time.add(completion_time.get(i) -
    process_clone.get(i).arrival_time);
```

```
		}
		for(int i=0; i<n; i++)
		{
			waiting_time.add(turn_around_time.get(i) -
process_clone.get(i).burst_time);
		}
		System.out.println("");
		System.out.println("****FCFS****");
		System.out.println("");
		System.out.println("N  AT BT CT  TAT WT");
		for(int i=0; i<n; i++)
		{
			System.out.print(process_clone.get(i).name + " ");
			System.out.print(process_clone.get(i).arrival_time + "  ");
			System.out.print(process_clone.get(i).burst_time + "  ");
			System.out.print(completion_time.get(i) + "   ");
			System.out.print(turn_around_time.get(i) + "   ");
			System.out.print(waiting_time.get(i));
			System.out.println("");
		}
		System.out.println("");
		System.out.println("Gant Chart: ");
		for(int i=1; i<gant_chart.size()+1; i++)
		{
			System.out.print(i + "  ");
		}
		System.out.println("");
		for(int i=0; i<gant_chart.size(); i++)
		{
			System.out.print(gant_chart.get(i) + " ");
		}
		System.out.println("");
		System.out.println("");
		float sum = 0;
		for(float t : turn_around_time)
		{
			sum += t;
		}
		float average_tat = sum/n;
		sum = 0;
		for(float t : waiting_time)
		{
			sum += t;
		}
		float average_wt = sum/n;
		System.out.println("Average turn around time: " + average_tat);
		System.out.println("Average waiting time: " + average_wt);
	}
}
```

Output:



```
C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>javac FCFS.java

C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>java MyClass
Enter number of processes:
3
Process name: p1
Arrival time for p1: 1
Burst time for p1: 3
Process name: p2
Arrival time for p2: 0
Burst time for p2: 5
Process name: p3
Arrival time for p3: 4
Burst time for p3: 4

****FCFS****

N  AT BT CT  TAT WT
p1 1  3  8   7   4
p2 0  5  5   5   0
p3 4  4  12   8   4

Gant Chart:
1  2  3  4  5  6  7  8  9  10  11  12
p2 p2 p2 p2 p2 p1 p1 p1 p3 p3 p3 p3

Average turn around time: 6.6666665
Average waiting time: 2.6666667

C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>
```

SJF:

```
import java.util.*;

import java.lang.*;

import java.io.*;


class Process

{

	String name;

	int arrival_time, burst_time;

	Process(String n, int at, int bt)

	{

		name = n;

		arrival_time = at;

		burst_time = bt;

	}
```

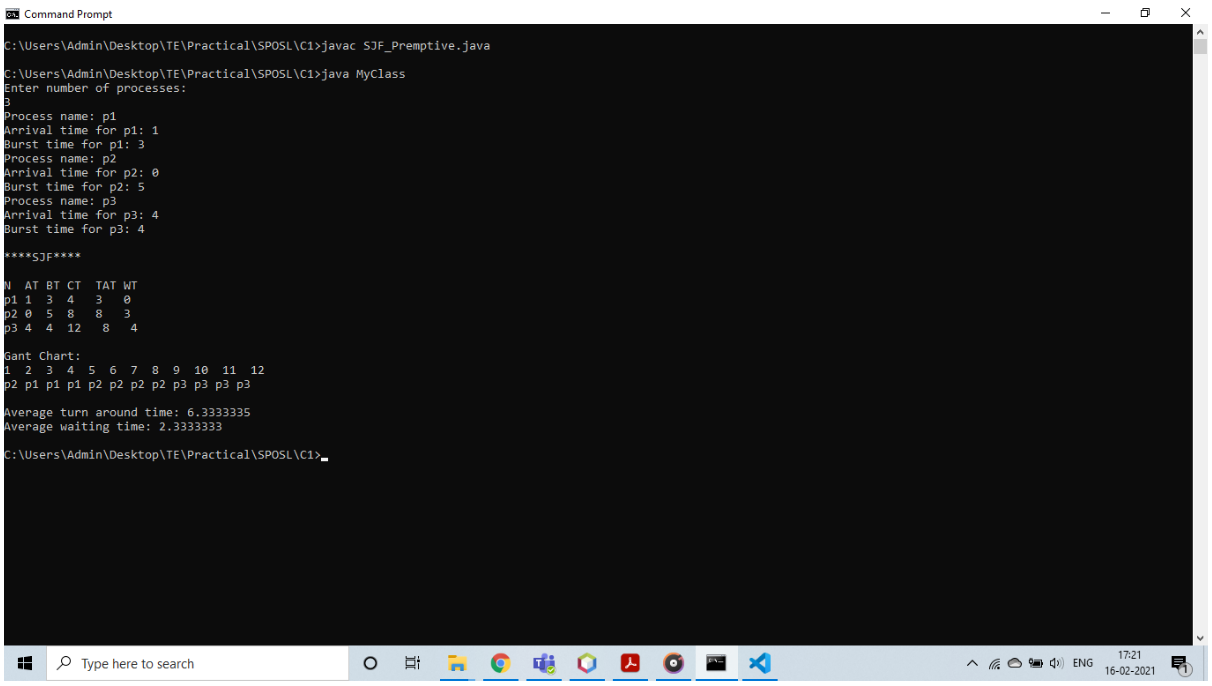
```
}

class SortByArrivalTime implements Comparator<Process>
{
        public int compare(Process a, Process b)
        {
                return a.arrival_time - b.arrival_time;
        }
}

class SortByBurstTime implements Comparator<Process>
{
        public int compare(Process a, Process b)
        {
                return a.burst_time - b.burst_time;
        }
}

class MyClass
{
        public static void main(String[] args) throws ClassCastException
        {
                Scanner sc = new Scanner(System.in);
                System.out.println("Enter number of processes: ");
                int n = sc.nextInt();
                Vector<Process> process = new Vector<Process>();
                Vector<Process> process_clone = new Vector<Process>();
                Vector<Integer> completion_time = new Vector<Integer>();
                Vector<Integer> turn_around_time = new Vector<Integer>();
```

```
		Vector<Integer> waiting_time = new Vector<Integer>();
		Vector<String> gant_chart = new Vector<String>();
		int bt, at;
		String name;
		for(int i=0; i<n; i++)
		{
			sc.nextLine();
			System.out.print("Process name: ");
			name = sc.nextLine();
			System.out.print("Arrival time for " + name + ": ");
			at = sc.nextInt();
			System.out.print("Burst time for " + name + ": ");
			bt = sc.nextInt();
			process.add(new Process(name, at, bt));
			process_clone.add(new Process(name, at, bt));
		}
		Collections.sort(process, new SortByArrivalTime());
		while(true)
		{
			int min_bt = 99999;
			int index = -1;
			for(int i=0; i<process.size(); i++)
			{
				if(process.get(i).burst_time < min_bt &&
process.get(i).arrival_time == 0)
				{
					index = i;
					min_bt = process.get(i).burst_time;
				}
			}
```

```
                }

                if(index != -1)
                {
                        process.get(index).burst_time--;
                        gant_chart.add(process.get(index).name);
                }
                else
                {
                        gant_chart.add("n");
                }
                for(int i=0; i<process.size(); i++)
                {
                        if(process.get(i).arrival_time != 0)
                        {
                                process.get(i).arrival_time--;
                        }
                }
                for(int i=0; i<process.size(); i++)
                {
                        if(process.get(i).burst_time == 0)
                        {
                                process.removeElementAt(i);
                        }
                }
                if(process.size() == 0)
                {
                        break;
                }
        }
        process = null;
```

```
                for(int i=0; i<n; i++)
                {
        completion_time.add(gant_chart.lastIndexOf(process_clone.get(i).name) + 1);
                }
                for(int i=0; i<n; i++)
                {
                        turn_around_time.add(completion_time.get(i) -
process_clone.get(i).arrival_time);
                }
                for(int i=0; i<n; i++)
                {
                        waiting_time.add(turn_around_time.get(i) -
process_clone.get(i).burst_time);
                }
                System.out.println("");
                System.out.println("****SJF****");
                System.out.println("");
                System.out.println("N  AT BT CT  TAT WT");
                for(int i=0; i<n; i++)
                {
                        System.out.print(process_clone.get(i).name + " ");
                        System.out.print(process_clone.get(i).arrival_time + "  ");
                        System.out.print(process_clone.get(i).burst_time + "  ");
                        System.out.print(completion_time.get(i) + "   ");
                        System.out.print(turn_around_time.get(i) + "   ");
                        System.out.print(waiting_time.get(i));
                        System.out.println("");
                }
                System.out.println("");
        }

                System.out.println("Gant Chart: ");
                for(int i=1; i<gant_chart.size()+1; i++)
                {
                        System.out.print(i + "  ");
                }
                System.out.println("");
                for(int i=0; i<gant_chart.size(); i++)
                {
                        System.out.print(gant_chart.get(i) + " ");
                }
                System.out.println("");
                System.out.println("");
                float sum = 0;
                for(float t : turn_around_time)
                {
                        sum += t;
                }
                float average_tat = sum/n;
                sum = 0;
                for(float t : waiting_time)
                {
                        sum += t;
                }
                float average_wt = sum/n;
                System.out.println("Average turn around time: " + average_tat);
                System.out.println("Average waiting time: " + average_wt);
        }
}
```

Output:



```
C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>javac SJF_Premptive.java

C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>java MyClass
Enter number of processes:
3
Process name: p1
Arrival time for p1: 1
Burst time for p1: 3
Process name: p2
Arrival time for p2: 0
Burst time for p2: 5
Process name: p3
Arrival time for p3: 4
Burst time for p3: 4

****SJF****

N  AT BT CT  TAT WT
p1 1  3  4   3   0
p2 0  5  8   8   3
p3 4  4  12   8   4

Gant Chart:
1  2  3  4  5  6  7  8  9  10  11  12
p2 p1 p1 p1 p2 p2 p2 p2 p3 p3 p3 p3

Average turn around time: 6.3333335
Average waiting time: 2.3333333

C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>
```

Round Robin:

```
import java.util.*;


class Process
{
        String name;
        int arrival_time, burst_time;
        Process(String n, int at, int bt)
        {
                name = n;
                arrival_time = at;
                burst_time = bt;
        }
}
```

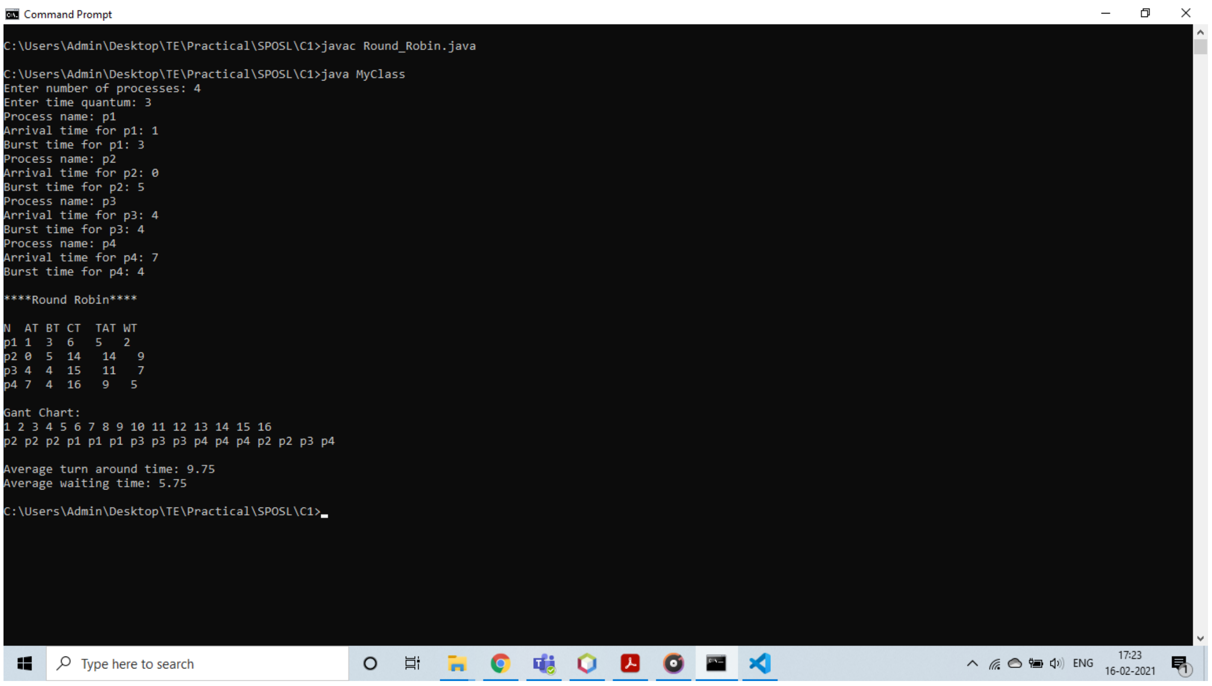
```
class SortByArrivalTime implements Comparator<Process>
{
        public int compare(Process a, Process b)
        {
                return a.arrival_time - b.arrival_time;
        }
}

class SortByBurstTime implements Comparator<Process>
{
        public int compare(Process a, Process b)
        {
                return a.burst_time - b.burst_time;
        }
}

class MyClass
{
  public static void main(String[] args)
  {
    Scanner sc = new Scanner(System.in);
                System.out.print("Enter number of processes: ");
                int n = sc.nextInt();
                Vector<Process> process = new Vector<Process>();
                Vector<Process> process_clone = new Vector<Process>();
                Vector<Integer> completion_time = new Vector<Integer>();
                Vector<Integer> turn_around_time = new Vector<Integer>();
                Vector<Integer> waiting_time = new Vector<Integer>();
                Vector<String> gant_chart = new Vector<String>();
```

```
int bt, at, time_quantum;
System.out.print("Enter time quantum: ");
time_quantum = sc.nextInt();
        String name;
        for(int i=0; i<n; i++)
        {
              sc.nextLine();
              System.out.print("Process name: ");
              name = sc.nextLine();
              System.out.print("Arrival time for " + name + ": ");
              at = sc.nextInt();
              System.out.print("Burst time for " + name + ": ");
              bt = sc.nextInt();
              process.add(new Process(name, at, bt));
              process_clone.add(new Process(name, at, bt));
        }
        Collections.sort(process_clone, new SortByArrivalTime());
//check if any process has arrival time = 0
//if not, wait till it becomes zero
if(process_clone.get(0).arrival_time != 0)
{
  int t = process_clone.get(0).arrival_time;
  for(int i=0; i<t; i++)
  {
    gant_chart.add("n");
  }
  for(int i=0; i<n; i++)
  {
    process_clone.get(i).arrival_time -= t;
    }
}
int temp = 0;
while(true)
{
    if(process_clone.size() == 0)
    {
        break;
    }
    for(int i=0; i<process_clone.size(); i++)
    {
        if(process_clone.get(i).arrival_time == 0)
        {
            if(process_clone.get(i).burst_time > time_quantum)
            {
                temp = time_quantum;
                process_clone.get(i).burst_time -= time_quantum;
                for(int j=0; j<time_quantum; j++)
                {
                    gant_chart.add(process_clone.get(i).name);
                }
            }
            else
            {
                temp = process_clone.get(i).burst_time;
                for(int j=0; j<process_clone.get(i).burst_time; j++)
                {
                    gant_chart.add(process_clone.get(i).name);
                }
```

```
                    process_clone.get(i).burst_time = 0;
                }
            }
            for(int j=0; j<process_clone.size(); j++)
            {
                if(process_clone.get(j).arrival_time < temp)
                {
                    process_clone.get(j).arrival_time = 0;
                }
                else
                {
                    process_clone.get(j).arrival_time -= temp;
                }
            }
        }
        for(int i=0; i<process_clone.size(); i++)
        {
            if(process_clone.get(i).burst_time == 0)
            {
                process_clone.removeElementAt(i);
            }
        }
    }
    for(int i=0; i<n; i++)
                  {
                              completion_time.add(gant_chart.lastIndexOf(process.get(i).name) +
1);
                  }
                  for(int i=0; i<n; i++)
                  {
                        turn_around_time.add(completion_time.get(i) -
process.get(i).arrival_time);
                }
                for(int i=0; i<n; i++)
                {
                        waiting_time.add(turn_around_time.get(i) -
process.get(i).burst_time);
                }
                System.out.println("");
                System.out.println("****Round Robin****");
                System.out.println("");
                System.out.println("N  AT BT CT  TAT WT");
                for(int i=0; i<n; i++)
                {
                        System.out.print(process.get(i).name + " ");
                        System.out.print(process.get(i).arrival_time + "  ");
                        System.out.print(process.get(i).burst_time + "  ");
                        System.out.print(completion_time.get(i) + "   ");
                        System.out.print(turn_around_time.get(i) + "   ");
                        System.out.print(waiting_time.get(i));
                        System.out.println("");
                }
                System.out.println("");
                System.out.println("Gant Chart: ");
                for(int i=1; i<gant_chart.size()+1; i++)
                {
                        System.out.print(i + " ");
                }
                System.out.println("");
                for(int i=0; i<gant_chart.size(); i++)
```

```
                    {
                            System.out.print(gant_chart.get(i) + " ");
                    }
                    System.out.println("");
                    System.out.println("");
                    float sum = 0;
                    for(float t : turn_around_time)
                    {
                            sum += t;
                    }
                    float average_tat = sum/n;
                    sum = 0;
                    for(float t : waiting_time)
                    {
                            sum += t;
                    }
                    float average_wt = sum/n;
                    System.out.println("Average turn around time: " + average_tat);
                    System.out.println("Average waiting time: " + average_wt);
    }
}
```

Output:



```
C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>javac Round_Robin.java

C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>java MyClass
Enter number of processes: 4
Enter time quantum: 3
Process name: p1
Arrival time for p1: 1
Burst time for p1: 3
Process name: p2
Arrival time for p2: 0
Burst time for p2: 5
Process name: p3
Arrival time for p3: 4
Burst time for p3: 4
Process name: p4
Arrival time for p4: 7
Burst time for p4: 4

****Round Robin****

N  AT BT CT  TAT WT
p1 1  3  6   5   2
p2 0  5  14   14   9
p3 4  4  15   11   7
p4 7  4  16   9   5

Gant Chart:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
p2 p2 p2 p1 p1 p1 p3 p3 p3 p4 p4 p4 p2 p2 p3 p4

Average turn around time: 9.75
Average waiting time: 5.75

C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>_
```

Priority Scheduling:

```
import java.util.*;

import java.lang.*;

import java.io.*;


class Process

{

        String name;

        int arrival_time, burst_time, priority;

        Process(String n, int at, int bt, int p)

        {

                name = n;

                arrival_time = at;

                burst_time = bt;

                priority = p;
```

```
	}
}

class SortByArrivalTime implements Comparator<Process>
{
	public int compare(Process a, Process b)
	{
		return a.arrival_time - b.arrival_time;
	}
}

class SortByBurstTime implements Comparator<Process>
{
	public int compare(Process a, Process b)
	{
		return a.burst_time - b.burst_time;
	}
}

class SortByPriority implements Comparator<Process>
{
	public int compare(Process a, Process b)
	{
		return a.priority - b.priority;
	}
}

class MyClass
{
```

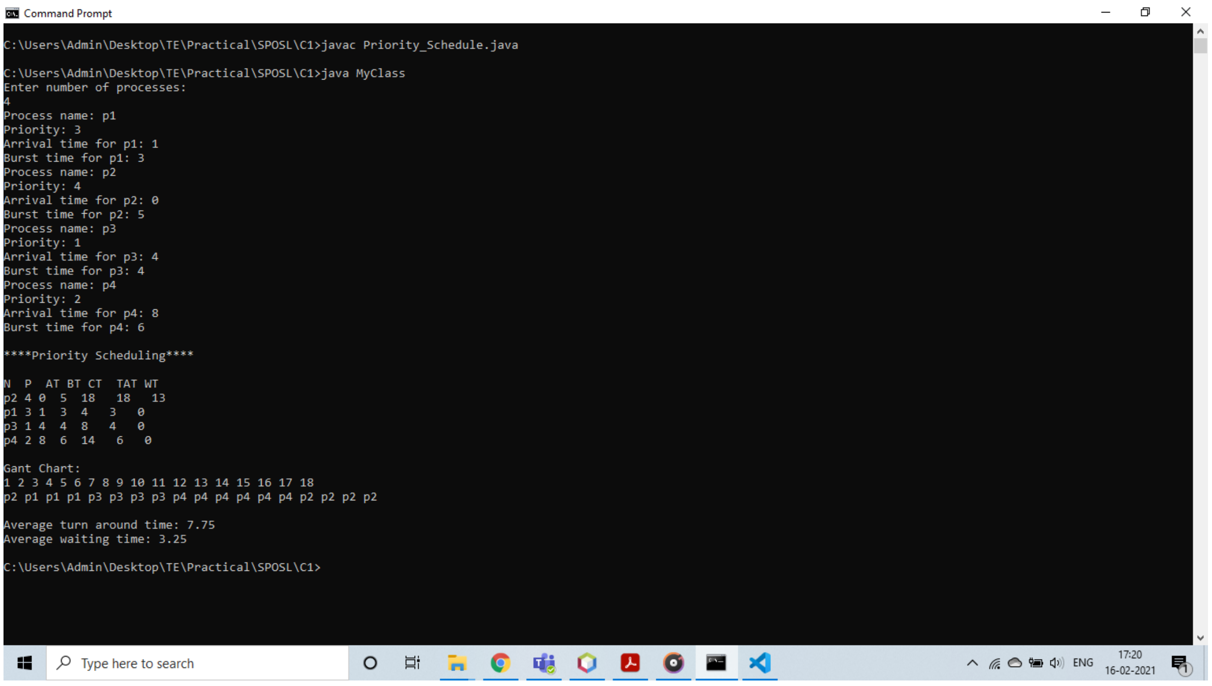
```
public static void main(String[] args) throws ClassCastException
{
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number of processes: ");
        int n = sc.nextInt();
        Vector<Process> process = new Vector<Process>();
        Vector<Process> process_clone = new Vector<Process>();
        Vector<Integer> completion_time = new Vector<Integer>();
        Vector<Integer> turn_around_time = new Vector<Integer>();
        Vector<Integer> waiting_time = new Vector<Integer>();
        Vector<String> gant_chart = new Vector<String>();
        int bt, at, priority;
        String name;
        for(int i=0; i<n; i++)
        {
                sc.nextLine();
                System.out.print("Process name: ");
                name = sc.nextLine();
                System.out.print("Priority: ");
                priority = sc.nextInt();
                System.out.print("Arrival time for " + name + ": ");
                at = sc.nextInt();
                System.out.print("Burst time for " + name + ": ");
                bt = sc.nextInt();
                process.add(new Process(name, at, bt, priority));
                process_clone.add(new Process(name, at, bt, priority));
        }
        Collections.sort(process_clone, new SortByArrivalTime());
        /*if(process.get(0).arrival_time != 0)
```

```
    {
        int t = process_clone.get(0).arrival_time;
        for(int i=0; i<t; i++)
        {
            gant_chart.add("n");
        }
        for(int i=0; i<n; i++)
        {
            process_clone.get(i).arrival_time -= t;
        }
                }*/
                while(true)
                {
                        int min_pt = 99999;
                        int index = -1;
                        for(int i=0; i<process.size(); i++)
                        {
                                if(process.get(i).priority < min_pt &&
process.get(i).arrival_time == 0)
                                {
                                        index = i;
                                        min_pt = process.get(i).priority;
                                }
                        }
                        if(index != -1)
                        {
                                process.get(index).burst_time--;
                                gant_chart.add(process.get(index).name);
                        }
                        else
                {
                        gant_chart.add("n");
                }
                for(int i=0; i<process.size(); i++)
                {
                        if(process.get(i).arrival_time != 0)
                        {
                                process.get(i).arrival_time--;
                        }
                }
                for(int i=0; i<process.size(); i++)
                {
                        if(process.get(i).burst_time == 0)
                        {
                                process.removeElementAt(i);
                        }
                }
                if(process.size() == 0)
                {
                        break;
                }
        }
        for(int i=0; i<n; i++)
        {

completion_time.add(gant_chart.lastIndexOf(process_clone.get(i).name) + 1);
        }
        for(int i=0; i<n; i++)
        {
```

```
				turn_around_time.add(completion_time.get(i) - 
process_clone.get(i).arrival_time);
			}
			for(int i=0; i<n; i++)
			{
				waiting_time.add(turn_around_time.get(i) - 
process_clone.get(i).burst_time);
			}
			System.out.println("");
			System.out.println("****Priority Scheduling****");
			System.out.println("");
			System.out.println("N  P  AT BT CT  TAT WT");
			for(int i=0; i<n; i++)
			{
				System.out.print(process_clone.get(i).name + " ");
				System.out.print(process_clone.get(i).priority + " ");
				System.out.print(process_clone.get(i).arrival_time + "  ");
				System.out.print(process_clone.get(i).burst_time + "  ");
				System.out.print(completion_time.get(i) + "   ");
				System.out.print(turn_around_time.get(i) + "   ");
				System.out.print(waiting_time.get(i));
				System.out.println("");
			}
			System.out.println("");
			System.out.println("Gant Chart: ");
			for(int i=1; i<gant_chart.size()+1; i++)
			{
				System.out.print(i + " ");
			}
			System.out.println("");
```

```
		for(int i=0; i<gant_chart.size(); i++)
		{
			System.out.print(gant_chart.get(i) + " ");
		}
		System.out.println("");
		System.out.println("");
		float sum = 0;
		for(float t : turn_around_time)
		{
			sum += t;
		}
		float average_tat = sum/n;
		sum = 0;
		for(float t : waiting_time)
		{
			sum += t;
		}
		float average_wt = sum/n;
		System.out.println("Average turn around time: " + average_tat);
		System.out.println("Average waiting time: " + average_wt);
	}
}
```

Output:



```
C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>javac Priority_Schedule.java

C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>java MyClass
Enter number of processes:
4
Process name: p1
Priority: 3
Arrival time for p1: 1
Burst time for p1: 3
Process name: p2
Priority: 4
Arrival time for p2: 0
Burst time for p2: 5
Process name: p3
Priority: 1
Arrival time for p3: 4
Burst time for p3: 4
Process name: p4
Priority: 2
Arrival time for p4: 8
Burst time for p4: 6

****Priority Scheduling****

N  P  AT BT CT  TAT WT
p2 4 0  5  18   18   13
p1 3 1  3  4   3   0
p3 1 4  4  8   4   0
p4 2 8  6  14   6   0

Gant Chart:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
p2 p1 p1 p1 p3 p3 p3 p3 p4 p4 p4 p4 p4 p4 p2 p2 p2 p2

Average turn around time: 7.75
Average waiting time: 3.25

C:\Users\Admin\Desktop\TE\Practical\SPOSL\C1>
```