UNIX and C Programming Lab Exercises

a) Rename a file using UNIX command

Aim:

To rename a file using a UNIX command.

Procedure:

1. Open terminal.
2. Use the mv command to rename the file.

Program:

```
#!/bin/bash
echo -n "Enter current filename: "
read oldname
echo -n "Enter new filename: "
read newname
mv "$oldname" "$newname"
echo "File renamed successfully."
```

Output:

```
Enter current filename: old.txt
Enter new filename: new.txt
File renamed successfully.
```

Result:

The file was renamed successfully using a shell script.

b) Area and Circumference of a Circle

Aim:

To write a shell program to calculate the area and circumference of a circle.

Procedure:

1. Accept radius from the user.
2. Use the formula:
   - Area = pir^2
   - Circumference = 2pir

3. Display results.

Program:

```
#!/bin/bash
echo -n "Enter radius: "
read r
area=$(echo "scale=2; 3.14159 * $r * $r" | bc)
circum=$(echo "scale=2; 2 * 3.14159 * $r" | bc)
echo "Area: $area"
echo "Circumference: $circum"
```

Output:

```
Enter radius: 5
Area: 78.54
Circumference: 31.42
```

Result:

Successfully calculated the area and circumference of the circle.

c) Priority Scheduling in C (Minimum Line Code)

Aim:

To implement Priority Scheduling algorithm in C using minimum lines.

Procedure:

1. Accept burst time and priority for each process.
2. Sort processes based on priority.
3. Calculate WT and TAT.
4. Display result.

Program:

```
#include <stdio.h>
int main() {
    int n, i, j, temp, bt[10], p[10], pr[10], wt=0, tat=0;
    printf("Enter no. of processes: ");
    scanf("%d", &n);
```

```
    for(i=0;i<n;i++){
        printf("BT and Priority for P%d: ",i+1);
        scanf("%d%d",&bt[i],&pr[i]);
        p[i]=i+1;
    }
    for(i=0;i<n-1;i++)
        for(j=i+1;j<n;j++)
            if(pr[i]>pr[j]) {
                temp=pr[i]; pr[i]=pr[j]; pr[j]=temp;
                temp=bt[i]; bt[i]=bt[j]; bt[j]=temp;
                temp=p[i]; p[i]=p[j]; p[j]=temp;
            }
    printf("P\tBT\tWT\tTAT\n");
    for(i=0;i<n;i++) {
        printf("P%d\t%d\t%d\t%d\n",p[i],bt[i],wt,wt+bt[i]);
        tat += wt + bt[i];
        wt += bt[i];
    }
    printf("Avg TAT: %.2f\n", (float)tat/n);
}
```

Output:

Enter no. of processes: 3

BT and Priority for P1: 5 2

BT and Priority for P2: 3 1

BT and Priority for P3: 4 3

P BT WT TAT

P2 3 0 3

P1 5 3 8

P3 4 8 12

Avg TAT: 7.67

Result:

Priority Scheduling was implemented using minimum lines in C.