Assignment 5

code:

```
#include<stdio.h>


void menu()
{
	printf("\n\n\t\t\t!!BANKERS ALGORITHM!!");
	printf("\n\n\t\t1.ACCEPT");
	printf("\n\n\t\t2.DISPLAY");
	printf("\n\n\t\t3.SAFE SEQUENCE");
	printf("\n\n\t\t4.ENTER REQUEST");
	printf("\n\n\t\t5.QUIT");
}

void accept(int a[][40],int p,int r)
{
	int i,j;
	for(i=0;i<p;i++)
	{
		for(j=0;j<r;j++)
		{
			printf("\nENTER RESOURCE R%d FOR PROCESS P%d : ",j,i);
			scanf("%d",&a[i][j]);
		}
	}
}

void display(int a[][40],int p,int r)
{
	int i,j;
	printf("\n\t");
	for(i=0;i<r;i++)
		printf("\tR%d ",i);

	for(i=0;i<p;i++)
	{
		printf("\n\n\tP%d ",i);
		for(j=0;j<r;j++)
		{
			printf("\t%d",a[i][j]);
		}
	}
}

void accept_total(int total[],int r)
```

```
{
	int i;
	for(i=0;i<r;i++)
	{
		printf("\nENTER R%d: ",i);
		scanf("%d",&total[i]);
	}
}

void disp_total(int total[],int r)
{
	int i;
	printf("\n\t");
	for(i=0;i<r;i++)
		printf("\tR%d ",i);

	printf("\n\n\t");
	for(i=0;i<r;i++)
	{
		printf("\t%d",total[i]);
	}

}

void cal_need(int all[][40],int max[][40],int need [][40],int p,int r)
{
	int i,j;
	for(i=0;i<p;i++)
	{
		for(j=0;j<r;j++)
		{
			need[i][j]=max[i][j]-all[i][j];
		}
	}
}

void cal_avail(int all[][40],int total[],int avail [],int p,int r)
{
	int i,j,sum;
	for(i=0;i<r;i++)
	{
		sum=0;
		for(j=0;j<p;j++)
		{
			sum=sum+all[j][i];
		}
		avail[i]=total[i]-sum;
	}
}
```

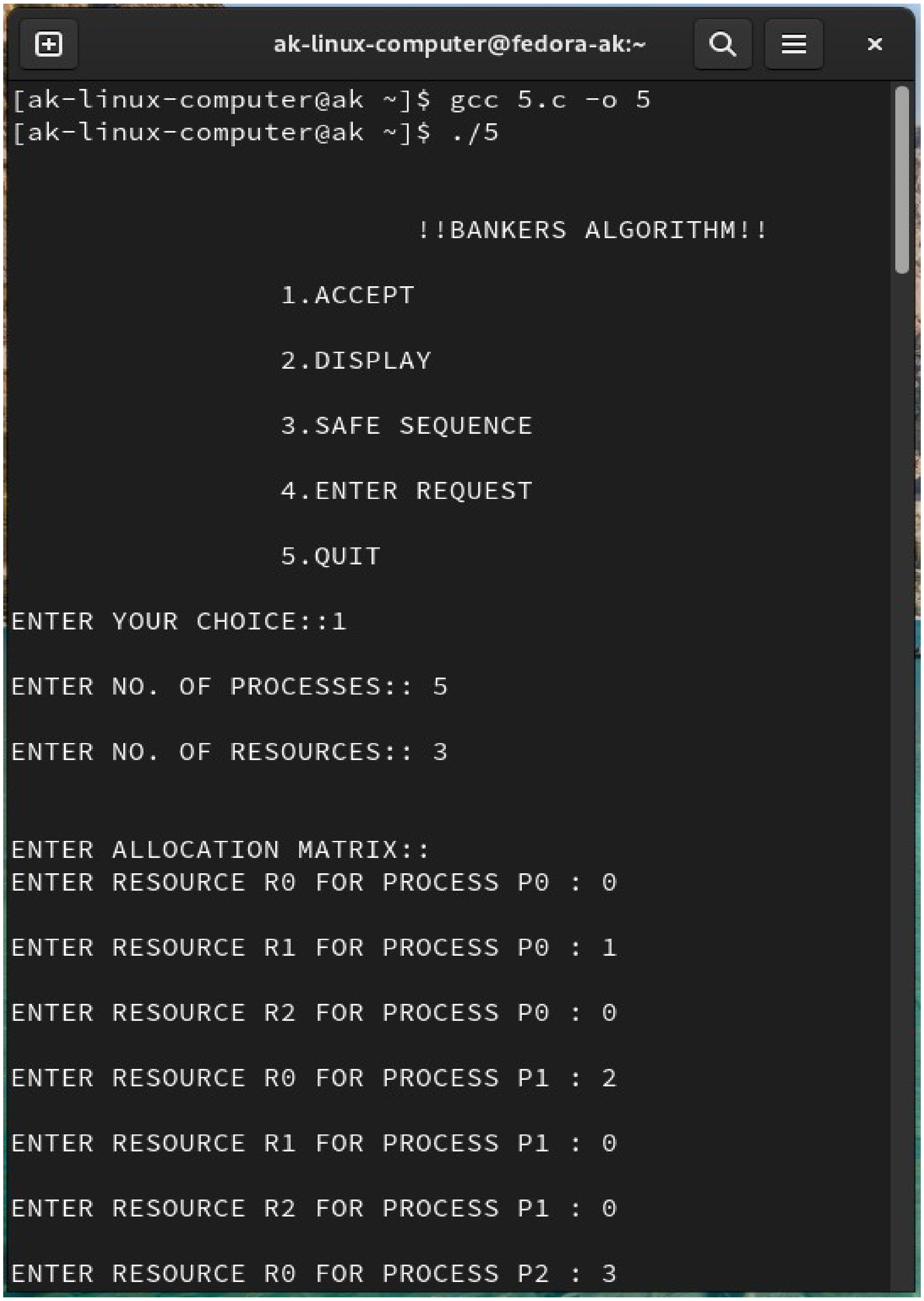
```
int safe_seq(int all[][40],int need[][40],int avail [],int p,int r)
{
        int seq[15],work[40],i,j,flag=0,k=0;
        int finish[20]={0};
        for(i=0;i<r;i++)
                work[i]=avail[i];

        while(flag==0)
        {
                flag=1;
                for(i=0;i<p;i++)
                {
                        if(finish[i]==0)
                        {
                                for(j=0;j<r;j++)
                                {
                                        if(need[i][j]>work[j])
                                                break;
                                }
                                if(j==r)
                                {
                                        finish[i]=1;
                                        for(j=0;j<r;j++)
                                                work[j]=work[j]+all[i][j];

                                        seq[k++]=i;     //store proc no. in seq
                                        flag=0;
                                }
                        }
                }
        }
        if(k==p)
        {
                printf("\n\nSYSTEM IS IN SAFE SEQUENCE & SAFE SEQUENCE IS::\n\n");
                for(i=0;i<k;i++)
                        printf("\tP%d",seq[i]);
                return 1;
        }
        return 0;
}

void request(int all[][40],int need[][40],int avail [],int p,int r)
{
        int dall[40][40],dneed[40][40],davail[40];
        int req[20],n,i,j;
        printf("\n\nENTER PROCESS NO.:: ");

        scanf("%d",&n);
```

```
	printf("\n\nENTER REQUEST:: ");
	for(i=0;i<r;i++)
	{
		printf("\nENTER R%d: ",i);

		scanf("%d",&req[i]);
	}
	for(i=0;i<r;i++)
	{
		if(req[i]>avail[i] || req[i]>need[n][i])
		{
			printf("\n\nREQUEST CANNOT BE GRANTED");
			return;
		}
	}
	for(i=0;i<p;i++)
	{
		for(j=0;j<r;j++)
		{
			dall[i][j]=all[i][j];
			dneed[i][j]=need[i][j];
		}
	}
	for(i=0;i<r;i++)
	{
		davail[i]=avail[i];
	}
	for(i=0;i<r;i++)
	{
		dall[n][i]=dall[n][i]+req[i];
		dneed[n][i]=dneed[n][i]-req[i];
		davail[i]=davail[i]-req[i];
	}
	if(safe_seq(dall,dneed,davail,p,r)==1)
		printf("\n\nREQUEST SHOULD BE GRANTED");
	else
		printf("\n\nREQUEST SHOULD NOT BE GRANTED");
}

int main()
{
	int all[40][40],need[40][40],max[40][40],p,r;
	int avail[40],total[40];
	int ch,x;

	while(ch!=5)
	{
		menu();
		printf("\n\nENTER YOUR CHOICE::");
```

```
                scanf("%d",&ch);
                switch(ch)
                {
                        case 1:
                                printf("\nENTER NO. OF PROCESSES:: ");
                                scanf("%d",&p);
                                printf("\nENTER NO. OF RESOURCES:: ");
                                scanf("%d",&r);
                                printf("\n\nENTER ALLOCATION MATRIX::");
                                accept(all,p,r);
                                printf("\n\nENTER MAXIMUM MATRIX::");
                                accept(max,p,r);
                                printf("\n\nENTER TATAL NO. OF RESOURCES:\n");
                                accept_total(total,r);
                                cal_need(all,max,need,p,r);
                                cal_avail(all,total,avail,p,r);
                                break;
                        case 2:
                                printf("\n\nALLOCATION MATRIX::");
                                display(all,p,r);
                                printf("\n\nMAXIMUM MATRIX::");
                                display(max,p,r);
                                printf("\n\nNEED MATRIX::");
                                display(need,p,r);
                                printf("\n\nTOTAL RESOURCES ARE:");
                                disp_total(total,r);
                                printf("\n\nTOTAL AVAILABLE RESOURCES ARE:");
                                disp_total(avail,r);
                                break;
                        case 3:
                                x=safe_seq(all,need,avail,p,r);
                                if(x==0)
                                        printf("\n\nSYSTEM IS NOT IN SAFE SEQUENCE");
                                else
                                        printf("\n");
                                break;
                        case 4:
                                request(all,need,avail,p,r);
                                break;

                }

        }
}
```

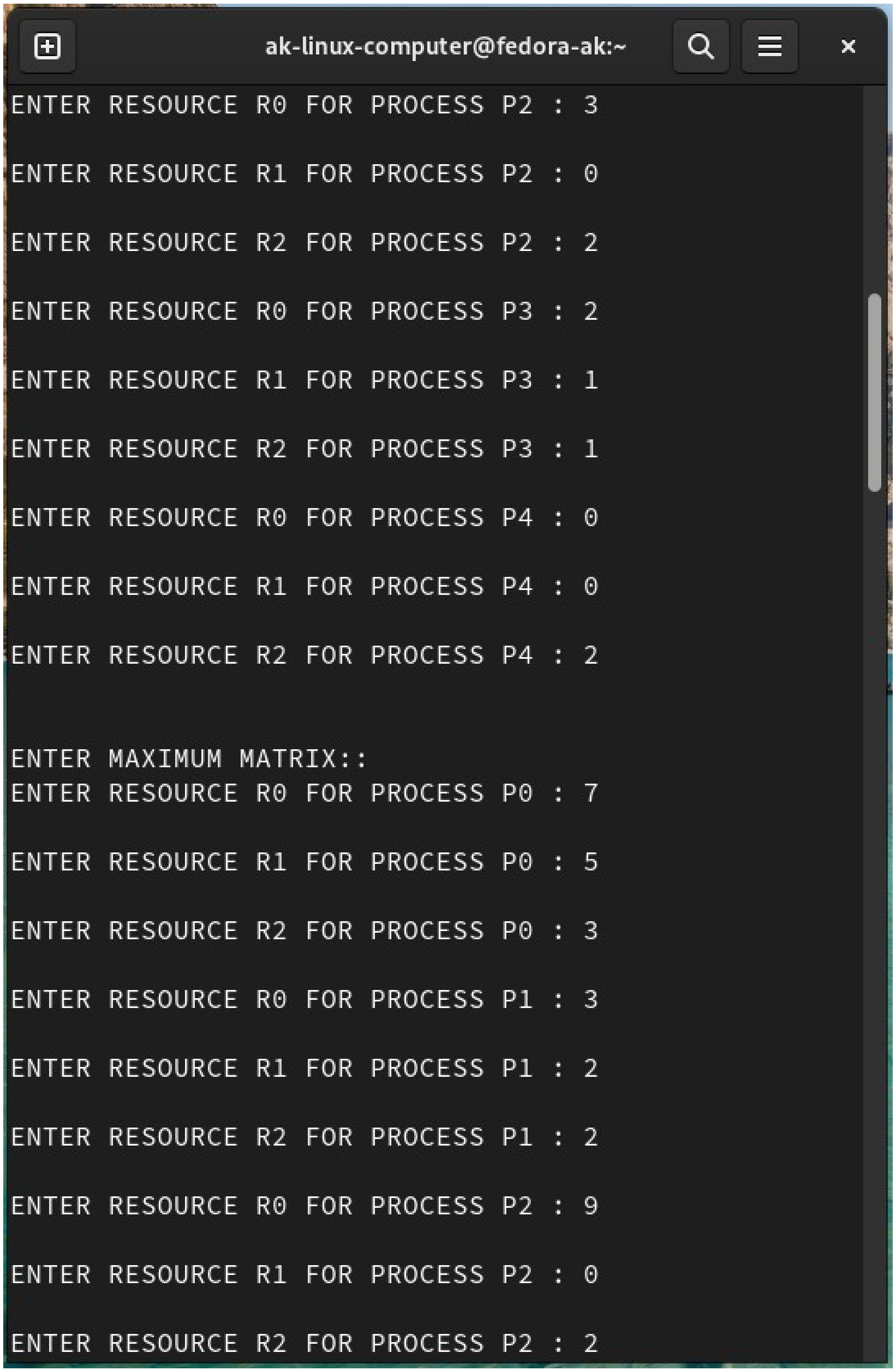
Output:

1:



```
[ak-linux-computer@ak ~]$ gcc 5.c -o 5
[ak-linux-computer@ak ~]$ ./5


                        !!BANKERS ALGORITHM!!

                1.ACCEPT

                2.DISPLAY

                3.SAFE SEQUENCE

                4.ENTER REQUEST

                5.QUIT

ENTER YOUR CHOICE::1

ENTER NO. OF PROCESSES:: 5

ENTER NO. OF RESOURCES:: 3


ENTER ALLOCATION MATRIX::
ENTER RESOURCE R0 FOR PROCESS P0 : 0

ENTER RESOURCE R1 FOR PROCESS P0 : 1

ENTER RESOURCE R2 FOR PROCESS P0 : 0

ENTER RESOURCE R0 FOR PROCESS P1 : 2

ENTER RESOURCE R1 FOR PROCESS P1 : 0

ENTER RESOURCE R2 FOR PROCESS P1 : 0

ENTER RESOURCE R0 FOR PROCESS P2 : 3
```

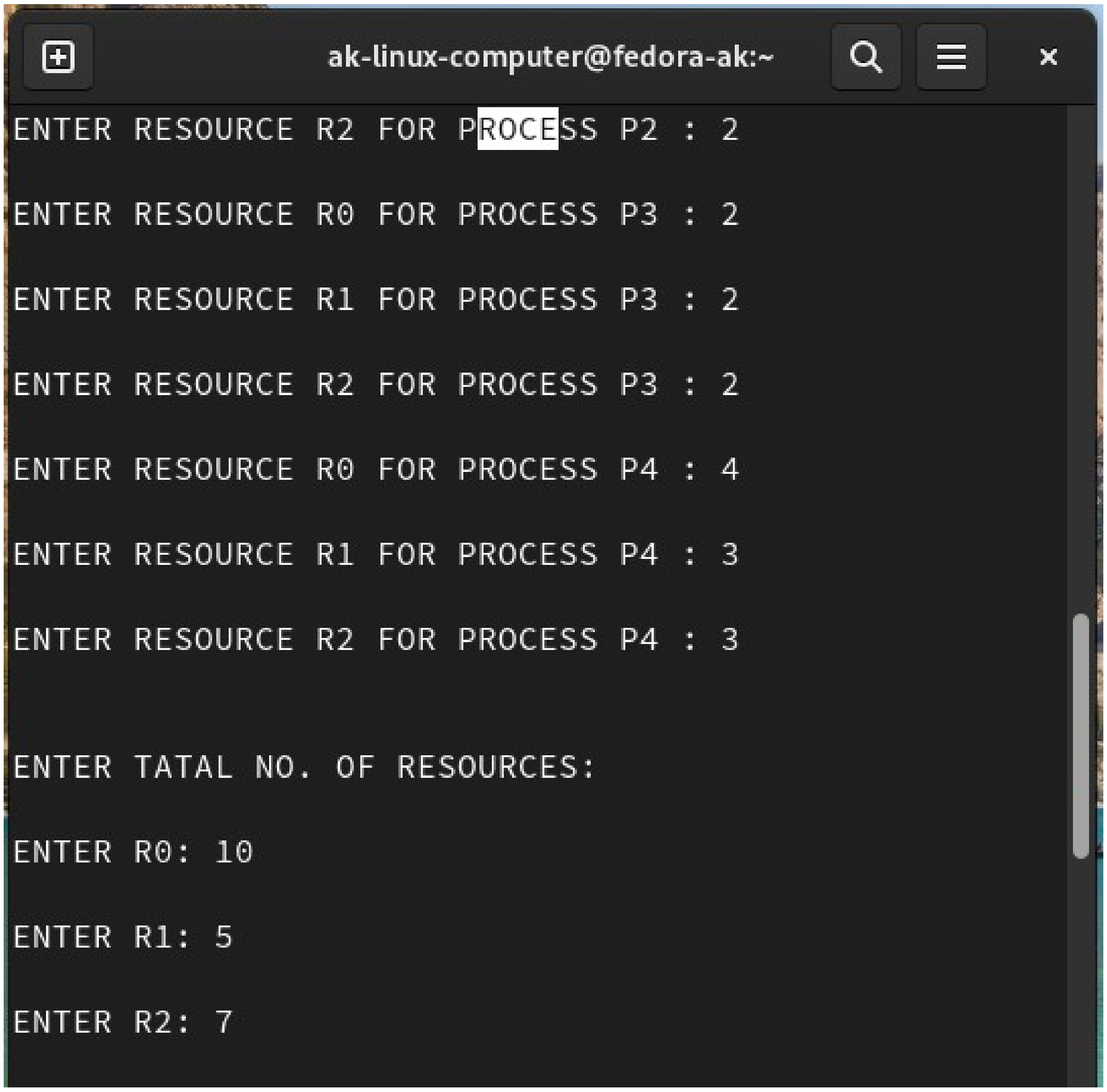
2:



ak-linux-computer@fedora-ak:~

```
ENTER RESOURCE R0 FOR PROCESS P2 : 3

ENTER RESOURCE R1 FOR PROCESS P2 : 0

ENTER RESOURCE R2 FOR PROCESS P2 : 2

ENTER RESOURCE R0 FOR PROCESS P3 : 2

ENTER RESOURCE R1 FOR PROCESS P3 : 1

ENTER RESOURCE R2 FOR PROCESS P3 : 1

ENTER RESOURCE R0 FOR PROCESS P4 : 0

ENTER RESOURCE R1 FOR PROCESS P4 : 0

ENTER RESOURCE R2 FOR PROCESS P4 : 2


ENTER MAXIMUM MATRIX::
ENTER RESOURCE R0 FOR PROCESS P0 : 7

ENTER RESOURCE R1 FOR PROCESS P0 : 5

ENTER RESOURCE R2 FOR PROCESS P0 : 3

ENTER RESOURCE R0 FOR PROCESS P1 : 3

ENTER RESOURCE R1 FOR PROCESS P1 : 2

ENTER RESOURCE R2 FOR PROCESS P1 : 2

ENTER RESOURCE R0 FOR PROCESS P2 : 9

ENTER RESOURCE R1 FOR PROCESS P2 : 0

ENTER RESOURCE R2 FOR PROCESS P2 : 2
```

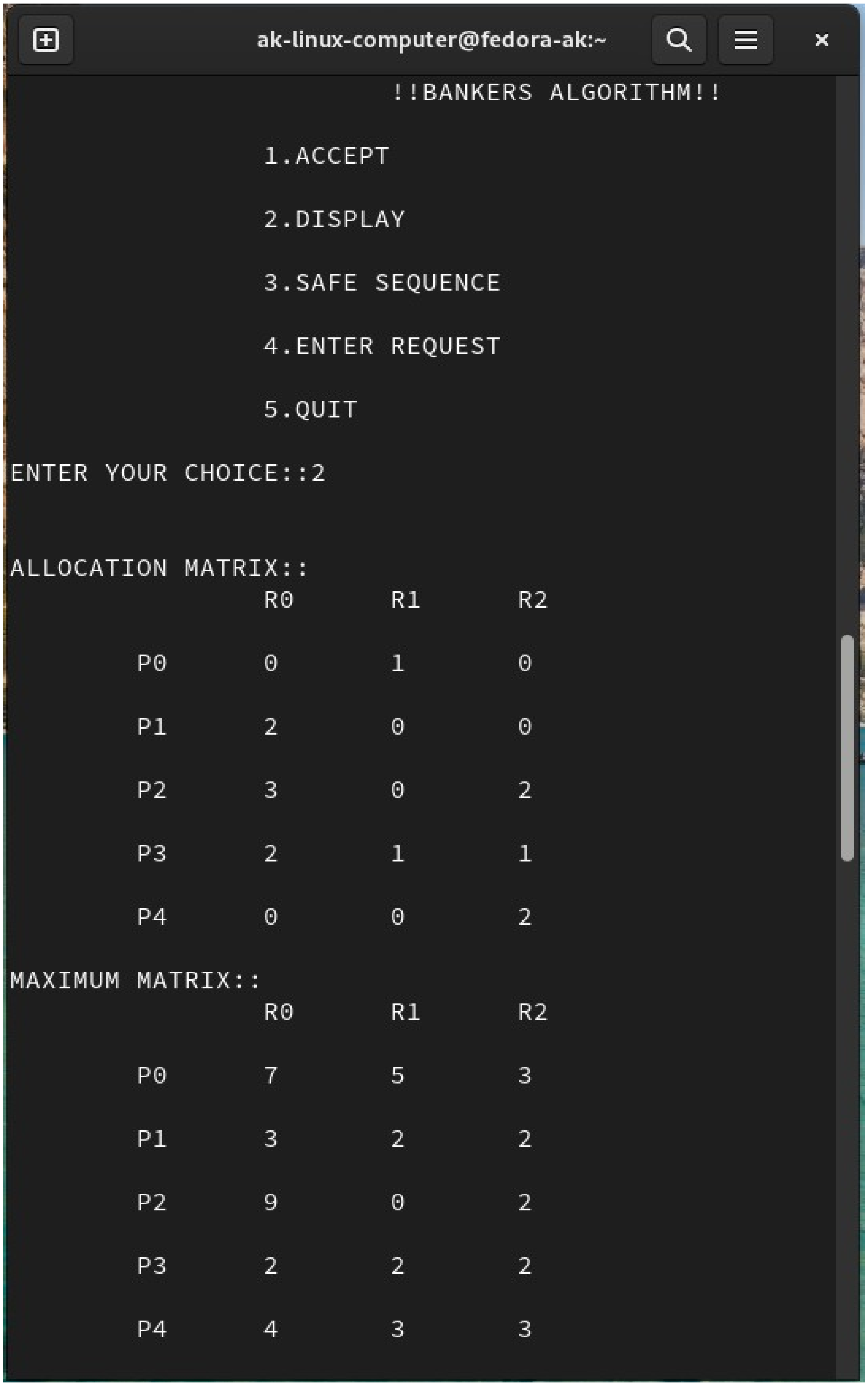
3:



ak-linux-computer@fedora-ak:~

```
ENTER RESOURCE R2 FOR PROCESS P2 : 2

ENTER RESOURCE R0 FOR PROCESS P3 : 2

ENTER RESOURCE R1 FOR PROCESS P3 : 2

ENTER RESOURCE R2 FOR PROCESS P3 : 2

ENTER RESOURCE R0 FOR PROCESS P4 : 4

ENTER RESOURCE R1 FOR PROCESS P4 : 3

ENTER RESOURCE R2 FOR PROCESS P4 : 3


ENTER TATAL NO. OF RESOURCES:

ENTER R0: 10

ENTER R1: 5

ENTER R2: 7
```

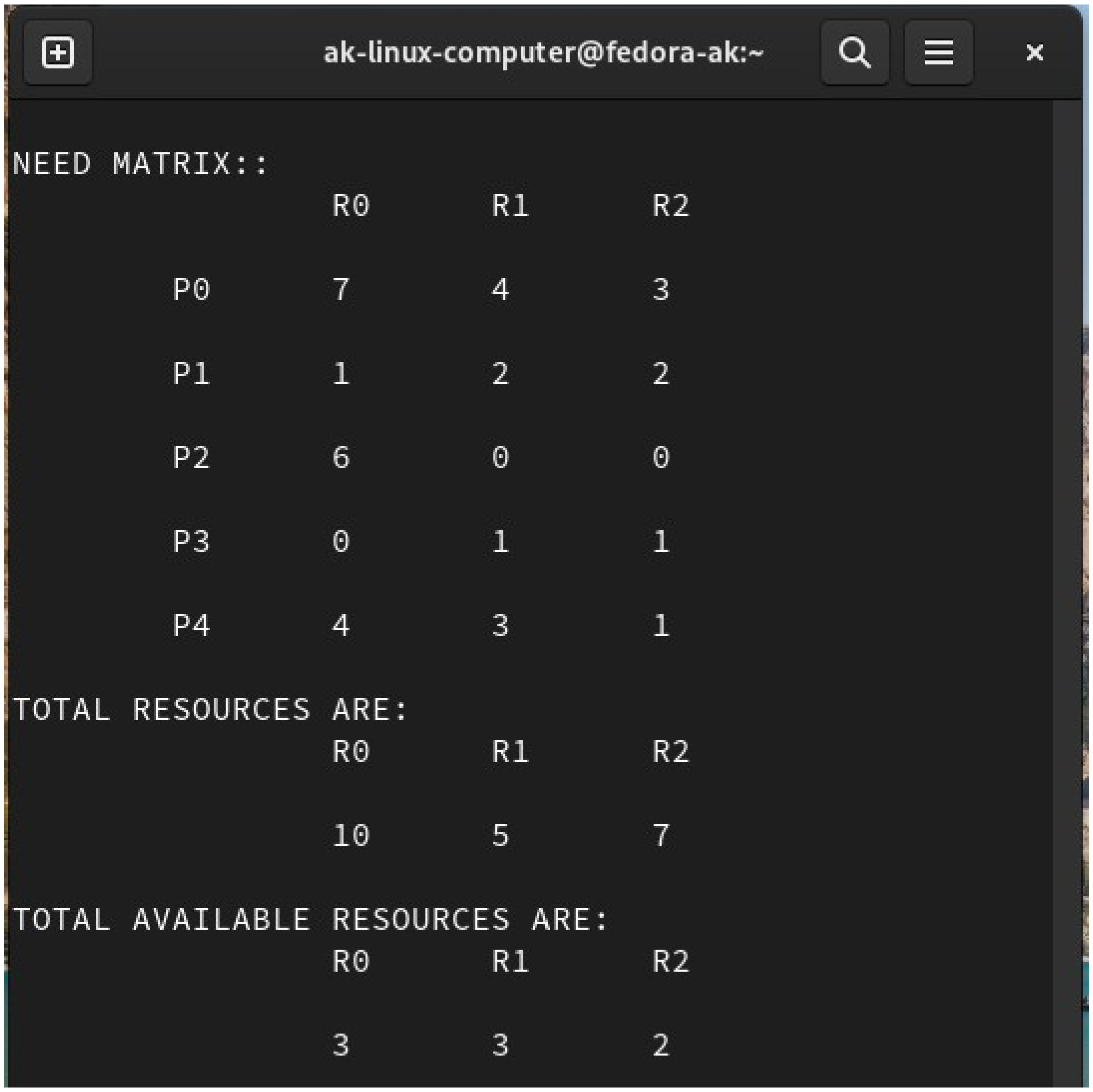
4:



ak-linux-computer@fedora-ak:~

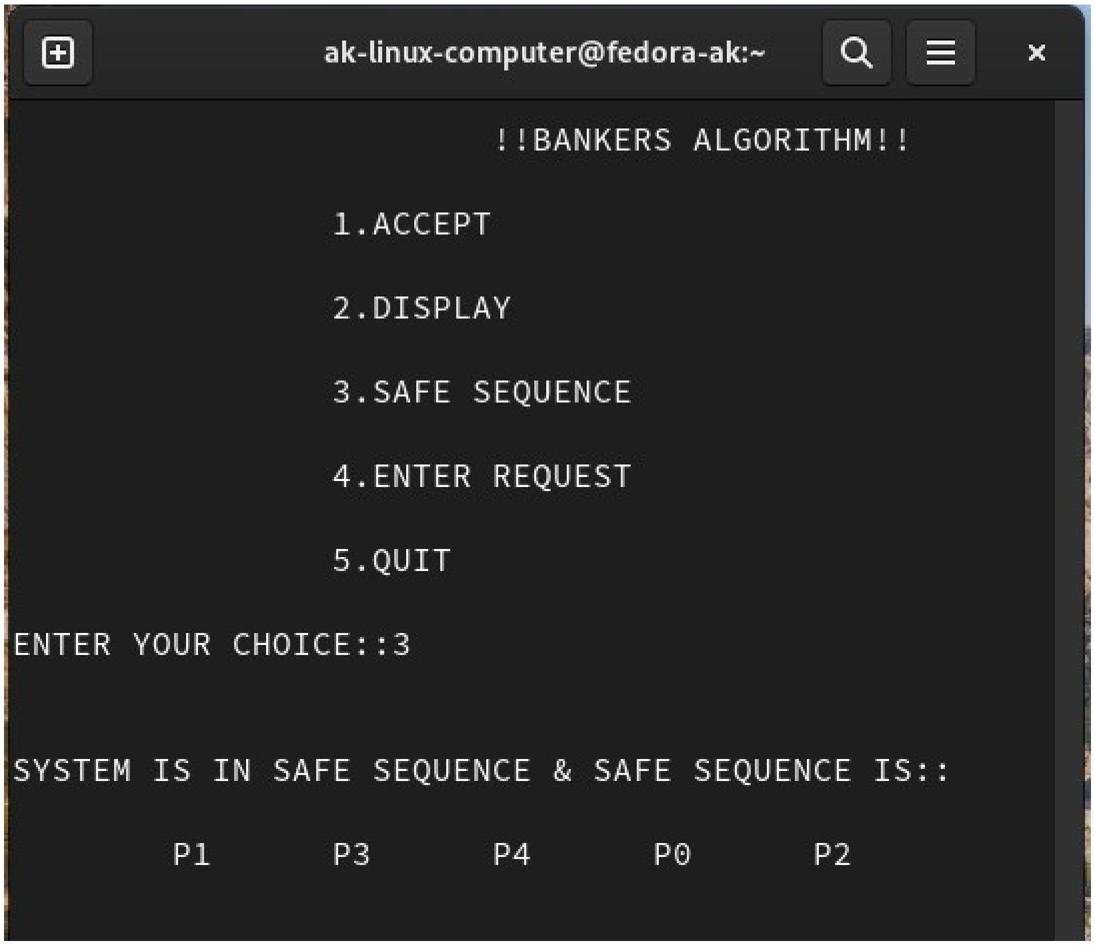
```
                        !!BANKERS ALGORITHM!!

                1.ACCEPT

                2.DISPLAY

                3.SAFE SEQUENCE

                4.ENTER REQUEST

                5.QUIT

ENTER YOUR CHOICE::2


ALLOCATION MATRIX::
                R0      R1      R2

        P0      0       1       0

        P1      2       0       0

        P2      3       0       2

        P3      2       1       1

        P4      0       0       2

MAXIMUM MATRIX::
                R0      R1      R2

        P0      7       5       3

        P1      3       2       2

        P2      9       0       2

        P3      2       2       2

        P4      4       3       3
```

5:



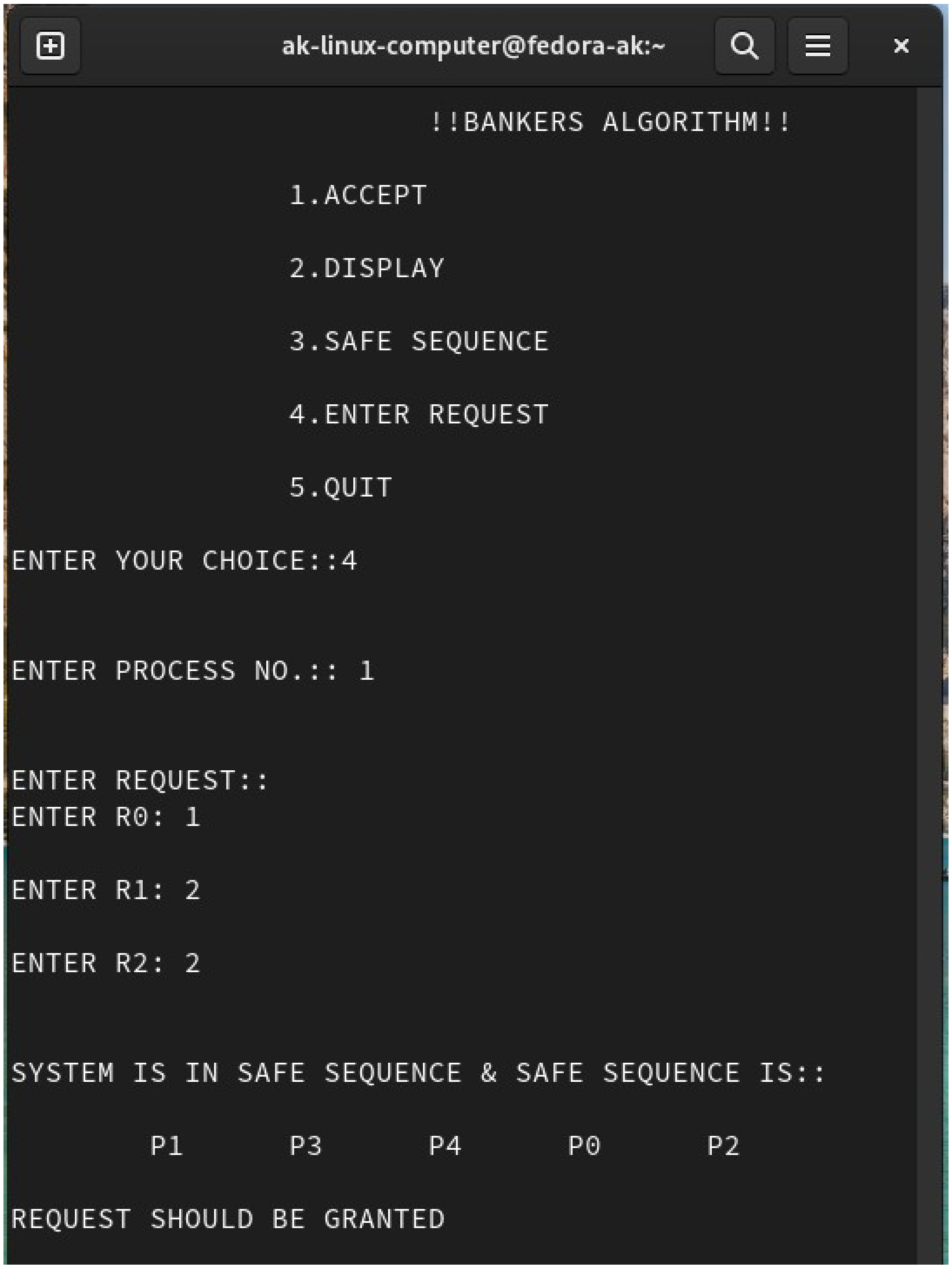
```
NEED MATRIX::
                R0      R1      R2

        P0      7       4       3

        P1      1       2       2

        P2      6       0       0

        P3      0       1       1

        P4      4       3       1

TOTAL RESOURCES ARE:
                R0      R1      R2

                10      5       7

TOTAL AVAILABLE RESOURCES ARE:
                R0      R1      R2

                3       3       2
```

6:



```
                        !!BANKERS ALGORITHM!!

                1.ACCEPT

                2.DISPLAY

                3.SAFE SEQUENCE

                4.ENTER REQUEST

                5.QUIT

ENTER YOUR CHOICE::3


SYSTEM IS IN SAFE SEQUENCE & SAFE SEQUENCE IS::

        P1      P3      P4      P0      P2
```

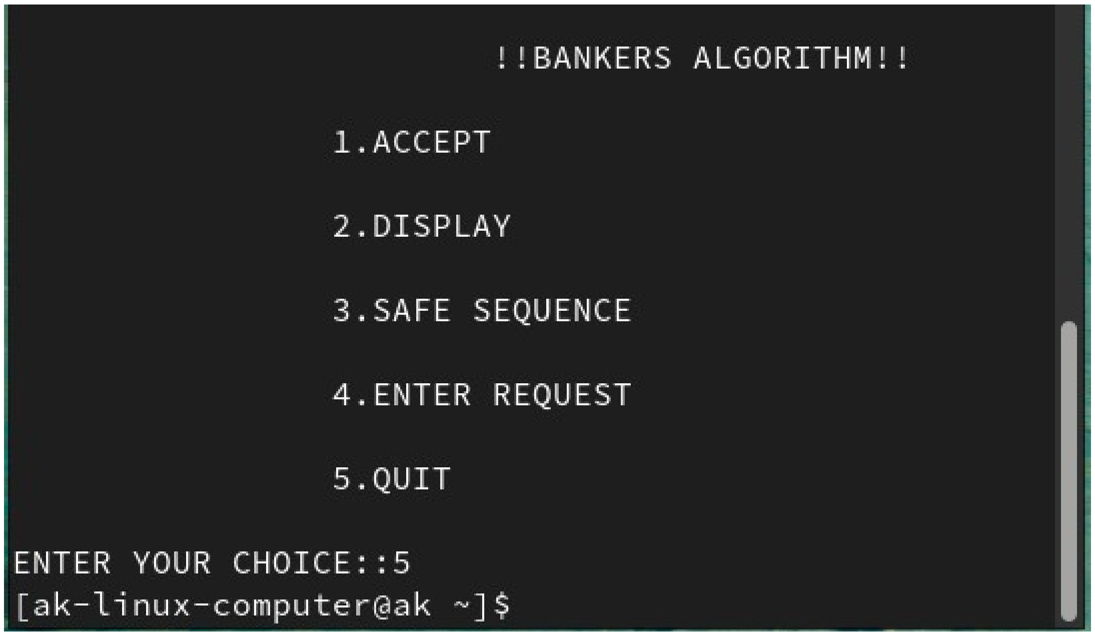
7:



ak-linux-computer@fedora-ak:~

```
                       !!BANKERS ALGORITHM!!

                1.ACCEPT

                2.DISPLAY

                3.SAFE SEQUENCE

                4.ENTER REQUEST

                5.QUIT

ENTER YOUR CHOICE::4


ENTER PROCESS NO.:: 1


ENTER REQUEST::
ENTER R0: 1

ENTER R1: 2

ENTER R2: 2


SYSTEM IS IN SAFE SEQUENCE & SAFE SEQUENCE IS::

        P1      P3      P4      P0      P2

REQUEST SHOULD BE GRANTED
```

8:



```
                        !!BANKERS ALGORITHM!!

                1.ACCEPT

                2.DISPLAY

                3.SAFE SEQUENCE

                4.ENTER REQUEST

                5.QUIT

ENTER YOUR CHOICE::5
[ak-linux-computer@ak ~]$
```