SLIP 1
Question.1

```
#include<stdio.h>
#define MAX 20
int frames[MAX],ref[MAX],mem[MAX][MAX],faults,
 sp,m,n,count[MAX];
void accept()
{
 int i;
 printf("Enter no.of frames:");
 scanf("%d", &n);
 printf("Enter no.of references:");
 scanf("%d", &m);
 printf("Enter reference string:\n");
 for(i=0;i<m;i++)
 {
  printf("[%d]=",i);
  scanf("%d",&ref[i]);
 }
}
void disp()
{
 int i,j;
 for(i=0;i<m;i++)
 printf("%3d",ref[i]);
 printf("\n\n");
 for(i=0;i<n;i++)
 {
  for(j=0;j<m;j++)
  {
   if(mem[i][j])
    printf("%3d",mem[i][j]);
   else
    printf("   ");
  }
  printf("\n");
 }
 printf("Total Page Faults: %d\n",faults);
}
int search(int pno)
{
 int i;
 for(i=0;i<n;i++)
 {
  if(frames[i]==pno)
   return i;
 }
 return -1;
}
int get_lfu(int sp)
{
 int i,min_i,min=9999;
 i=sp;
 do
 {
  if(count[i]<min)
```

Edit with WPS Office

```
    {
      min = count[i];
      min_i = i;
    }
    i=(i+1)%n;
  }while(i!=sp);
  return min_i;
}
void lfu()
{
  int i,j,k;
    for(i=0;i<m && sp<n;i++)
  {
    k=search(ref[i]);
    if(k==-1)
    {
      frames[sp]=ref[i];
      count[sp]++;
      faults++;
      sp++;
      for(j=0;j<n;j++)
        mem[j][i]=frames[j];
    }
    else
      count[k]++;
      }
  sp=0;
  for(;i<m;i++)
  {
    k = search(ref[i]);
    if(k==-1)
    {
      sp = get_lfu(sp);
      frames[sp] = ref[i];
      count[sp]=1;
      faults++;
      sp = (sp+1)%n;
    for(j=0;j<n;j++)
        mem[j][i] = frames[j];
    }
    else
      count[k]++;
  }
}
  int main()
{
  accept();
  lfu();
  disp();
  return 0;
}
```

SLIP 1 Question 2

```
#include <stdio.h>
#include <stdlib.h>
int count(char *fname);
```

Edit with WPS Office

```
int print(char *fname);
int top(char *fname, int c);
int bottom(char *fname, int c, int total);
int total, c, current;

void main(int argc, char *argv[])
{
   char *p;
   if (argc != 4)
      printf("invalid number of arguments");
   if (*argv[2] == '+' || *argv[2] == '-')
   {
      total = count(argv[3]);
      p = argv[2];
      p++;
      c = atoi(p);
      if (c > total)
         printf("invalid line count\n");
      if (*argv[2] == '+')
         top(argv[3], c);
      else if (*argv[2] == '-')
         bottom(argv[3], c, total);
      else
         print(argv[3]);
   }
}

int count(char *fname)
{
   int total, ch;
   FILE *fp;
   fp = fopen(fname, "r");
   if (fp == NULL)
   {
      printf("\nunable to open file");
      return (-1);
   }
   total = 0;
   ch = fgetc(fp);
   while (ch != EOF)
   {
      if (ch == 10)
         total++;
      ch = fgetc(fp);
   }
   total++;
   fclose(fp);
   return (total);
}

int print(char *fname)
{
   int ch;
   FILE *fp;
   fp = fopen(fname, "r");
   ch = fgetc(fp);
   while (ch != EOF)
```

Edit with WPS Office

```
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}

int top(char *fname, int c)
{
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (current < c)
    {
        printf("%c", ch);
        if (ch == 10)
            current++;
        ch = fgetc(fp);
    }
    fclose(fp);
}

int bottom(char *fname, int c, int total)
{
    int current;
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (1)
    {
        if (ch == 10)
            current++;
        if (current >= total - c)
            break;
        ch = fgetc(fp);
    }
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}
```

SLIP.2
Question 1

```
#include<stdio.h>
int RefString[10],PT[3];
void Accept()
{
            int i;
```

Edit with WPS Office

```
        printf("Enter Reference String:\n");
        for(i=0;i<10;i++)
        {
                printf("[%d]=",i);
                scanf("%d",&RefString[i]);
        }
}
int Search(int s)
{
        int i;
          for(i=0;i<3;i++)
                  if(PT[i]==s)
                          return(i);
          else return(-1);
}
void FIFO()
{
        int i,j,k=0,Faults=0;
          for(i=0;i<10;i++)
          {
                printf("%2d",RefString[i]);
                if(Search(RefString[i])==-1)
                {
                   PT[k]=RefString[i];
                    for(j=0;j<3;j++)
                    {
                        if(PT[j])
                        {
                        printf("%2d",PT[j]);
                        }
                    }
                        printf("\n");
                        Faults++;
                        k=(k+1)%3;
                  }
                    printf("\n");
              }
              printf("Total Page Faults:%d",Faults);
}

int main()
{
        Accept();

        FIFO();
        return 0;
}
```

SLIP 2 Question..2

count c filename

count w filename

count l filename

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>
```

Edit with WPS Office

```
#include <stdlib.h>
#include <string.h>
void make_toks(char *s, char *tok[])
{
int i=0;
char *p;
p = strtok(s," ");
while(p!=NULL)
{
  tok[i++]=p;
  p=strtok(NULL," ");
}
tok[i]=NULL;
}
void count(char *fn, char op)
{
int fh,cc=0,wc=0,lc=0;
char c;
fh = open(fn,O_RDONLY);
if(fh==-1)
{
  printf("File %s not found.\n",fn);
return;
}
while(read(fh,&c,1)>0)
{
if(c==' ') wc++;
else if(c=='\n')
  {
  wc++;
  lc++;
  }
  cc++;
}
close(fh);
switch(op)
{
case 'c':
  printf("No.of characters:%d\n",cc-1);
break;
case 'w':
  printf("No.of words:%d\n",wc);
break;
case 'l':
  printf("No.of lines:%d\n",lc+1);
break;
}
}
int main()
{
char buff[80],*args[10];
int pid;
while(1)
{
  printf("myshell$ ");
  fflush(stdin);
  fgets(buff,80,stdin);
```

Edit with WPS Office

```
  buff[strlen(buff)-1]='\0';
  make_toks(buff,args);
if(strcmp(args[0],"count")==0)
  count(args[2],args[1][0]);
else
  {
  pid = fork();
  if(pid>0)
  wait();
  else
  {
  if(execvp(args[0],args)==-1)
  printf("Bad command.\n");
  }
  }
}
return 0;
}
```

SLIP 3
Question 1

```
#include<stdio.h>
#define MAX 20
int fr[MAX],r[MAX],me[MAX][MAX],f,s,m,n,t[MAX];
void accept()
{
int i;
printf("enter no of frames:");
scanf("%d",&n);
printf("enter length of reference string:");
scanf("%d",&m);
printf("enter string elements:\n");
for(i=0;i<m;i++)
{
   printf("[%d]=",i );
   scanf("%d",&r[i]);
  }
}
void display()
{
int i,j;
for(i=0;i<m;i++)
     printf("%3d",r[i]);
  printf("\n");
for(i=0;i<n;i++)
  {
     for(j=0;j<m;j++)
     {
        if(me[i][j])
           printf("%3d",me[i][j]);
        else
           printf("       ");
     }
     printf("\n");
  }
  printf("total page faults:%d",f);
```

Edit with WPS Office

```
}
int search(int p)
{
int i;
for(i=0;i<n;i++)
   if(fr[i]==p)
      return 1;
return -1;
}
int get()
{
int i,mi,min=9999;
for(i=0;i<n;i++)
   if(t[i]<min)
   {
       min=t[i];
       mi=i;
    }
 return mi;
}
void lru()
{
int i,j,k;
for(i=0;i<m && s<n;i++)
{
   k=search(r[i]);
   if(k==-1)
   {
      fr[s]=r[i];
      t[s]=i;
      f++;
      s++;
      for(j=0;j<n;j++)
         me[j][i]=fr[j];
    }
   else
      t[k]=i;
 }
 for(i=0;i<m;i++)
 {
   k=search(r[i]);
   if(k==-1)
   {
      s=get();
      fr[s]=r[i];
      t[s]=i;
      f++;
      for(j=0;j<n;j++)
         me[j][i]=fr[j];
    }
    else
      t[k]=i;
   }
   }
   int main()
   {
   accept();
```

Edit with WPS Office

```
   lru();
   display();
   return 0;
   }
```

SLIP 3 Question 2

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void make_toks(char *s, char *tok[])
{
int i=0;
char *p;
p = strtok(s," ");
while(p!=NULL)
{
  tok[i++]=p;
  p=strtok(NULL," ");
}
tok[i]=NULL;
}
void count(char *fn, char op)
{
int fh,cc=0,wc=0,lc=0;
char c;
fh = open(fn,O_RDONLY);
if(fh==-1)
{
  printf("File %s not found.\n",fn);
return;
}
while(read(fh,&c,1)>0)
{
if(c==' ') wc++;
else if(c=='\n')
  {
  wc++;
  lc++;
  }
  cc++;
}
close(fh);
switch(op)
{
case 'c':
  printf("No.of characters:%d\n",cc-1);
break;
case 'w':
  printf("No.of words:%d\n",wc);
break;
case 'l':
  printf("No.of lines:%d\n",lc+1);
break;
```

Edit with WPS Office

```
}
}
int main()
{
char buff[80],*args[10];
int pid;
while(1)
{
  printf("myshell$ ");
  fflush(stdin);
  fgets(buff,80,stdin);
  buff[strlen(buff)-1]='\0';
  make_toks(buff,args);
if(strcmp(args[0],"count")==0)
  count(args[2],args[1][0]);
else
  {
  pid = fork();
  if(pid>0)
  wait();
  else
  {
  if(execvp(args[0],args)==-1)
  printf("Bad command.\n");
  }
  }
}
return 0;
}
```

*SLIP 5
Question 1

```
#include<stdio.h>
int n,page[20],f,fr[20],i;
void display()
        {
                for(i=0;i<f;i++)
                 {
                    printf("%d",fr[i]);
                 }
                      printf("\n");
        }
void request()
{
        printf("enter no.of pages:");
        scanf("%d",&n);
        printf("enter no.of frames:");
        scanf("%d",&f);
        printf("enter no.of page no:");
          for(i=0;i<n;i++)
          {
          scanf("%d",&page[i]);
          }
        for(i=0;i<n;i++)
```

Edit with WPS Office

```
        {
            fr[i]=-1;
        }
}
void replace()
{
        int j,flag=0,pf=0;
        int max,lp[10],index,m;
        for(j=0;j<f;j++)
        {
           fr[j]=page[j];
           flag=1;
           pf++;
display();
}
for(j=f;j<n;j++)
{
flag=0;
for(i=0;i<f;i++)
{
if(fr[i]==page[j])
{
flag=1;
break;
}
}
if(flag==0)
{
for(i=0;i<f;i++)
lp[i]=0;
for(i=0;i<f;i++)
{
for(m=j+1;m<n;m++)
{
if(fr[i]==page[m])
{
lp[i]=m-j;
break;
}
}
}
max=lp[0];
index=0;
for(i=0;i<f;i++)
{
if(lp[i]==0)
{
index=i;
break;
}
else
{
if(max<lp[i])
{
max=lp[i];
index=i;
}
```

Edit with WPS Office

```
}
}
fr[index]=page[j];
pf++;
display();
}
}
printf("page faults:%d",pf);
}
void main()
{
request();
replace();
}
```

SLIP 5 Question.2

```
#include <stdio.h>
#include <stdlib.h>
int count(char *fname);
int print(char *fname);
int top(char *fname, int c);
int bottom(char *fname, int c, int total);
int total, c, current;

void main(int argc, char *argv[])
{
   char *p;
   if (argc != 4)
      printf("invalid number of arguments");
   if (*argv[2] == '+' || *argv[2] == '-')
   {
      total = count(argv[3]);
      p = argv[2];
      p++;
      c = atoi(p);
      if (c > total)
         printf("invalid line count\n");
      if (*argv[2] == '+')
         top(argv[3], c);
      else if (*argv[2] == '-')
         bottom(argv[3], c, total);
      else
         print(argv[3]);
   }
}

int count(char *fname)
{
   int total, ch;
   FILE *fp;
   fp = fopen(fname, "r");
   if (fp == NULL)
   {
      printf("\nunable to open file");
      return (-1);
   }
```

Edit with WPS Office

```
    total = 0;
    ch = fgetc(fp);
    while (ch != EOF)
    {
        if (ch == 10)
            total++;
        ch = fgetc(fp);
    }
    total++;
    fclose(fp);
    return (total);
}

int print(char *fname)
{
    int ch;
    FILE *fp;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}

int top(char *fname, int c)
{
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (current < c)
    {
        printf("%c", ch);
        if (ch == 10)
            current++;
        ch = fgetc(fp);
    }
    fclose(fp);
}

int bottom(char *fname, int c, int total)
{
    int current;
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (1)
    {
        if (ch == 10)
            current++;
        if (current >= total - c)
```

Edit with WPS Office

```
            break;
        ch = fgetc(fp);
    }
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}
```

SLIP 10

Question.1

```
#include<stdio.h>
int RefString[15],PT[3];

void Accept()
{
            int i;
            printf("Enter Reference String:\n");
            for(i=0;i<10;i++)
            {
            printf("(%d)=",i);
            scanf("%d",&RefString[i]);
            }
}

int search(int s)
{
     int i;
     for(i=0;i<3;i++)
            if(PT[i]==s)
                return(i);
     return(-1);
}

void FIFO()
{
         int i,j,k=0,Faults=0;
         for(i=0;i<10;i++)
         {
                printf("%2d",RefString[i]);
                if(search(RefString[i])==-1)
                {
                            PT[k]=RefString[i];
                            for(j=0;j<3;j++)
                            {
                                        if(PT[j])
                                        {
                                        printf("%2d",PT[j]);
                                        }
                            }
                            printf("\n");
                            Faults++;
                            k=(k+1)%3;
```

Edit with WPS Office

```
            }
            printf("\n");

     }
     printf("Total Page Faults:%d",Faults);
}
   int main()
   {
            Accept();
            FIFO();
            return 0;
   }
}
```

SLIP 10
Question 2

```
#include <stdio.h>
#include <stdlib.h>
int count(char *fname);
int print(char *fname);
int top(char *fname, int c);
int bottom(char *fname, int c, int total);
int total, c, current;

void main(int argc, char *argv[])
{
   char *p;
   if (argc != 4)
      printf("invalid number of arguments");
   if (*argv[2] == '+' || *argv[2] == '-')
   {
      total = count(argv[3]);
      p = argv[2];
      p++;
      c = atoi(p);
      if (c > total)
         printf("invalid line count\n");
      if (*argv[2] == '+')
         top(argv[3], c);
      else if (*argv[2] == '-')
         bottom(argv[3], c, total);
      else
         print(argv[3]);
   }
}

int count(char *fname)
{
   int total, ch;
   FILE *fp;
   fp = fopen(fname, "r");
   if (fp == NULL)
   {
      printf("\nunable to open file");
      return (-1);
   }
```

Edit with WPS Office

```
    total = 0;
    ch = fgetc(fp);
    while (ch != EOF)
    {
        if (ch == 10)
            total++;
        ch = fgetc(fp);
    }
    total++;
    fclose(fp);
    return (total);
}

int print(char *fname)
{
    int ch;
    FILE *fp;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}

int top(char *fname, int c)
{
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (current < c)
    {
        printf("%c", ch);
        if (ch == 10)
            current++;
        ch = fgetc(fp);
    }
    fclose(fp);
}

int bottom(char *fname, int c, int total)
{
    int current;
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (1)
    {
        if (ch == 10)
            current++;
        if (current >= total - c)
```

Edit with WPS Office

```
            break;
        ch = fgetc(fp);
    }
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}
```

SLIP 12
Question 1

```
#include<stdio.h>
#define MAX 20
int fr[MAX],r[MAX],me[MAX][MAX],f,s,m,n,t[MAX];
void accept()
{
int i;
printf("enter no of frames:");
scanf("%d",&n);
printf("enter length of reference string:");
scanf("%d",&m);
printf("enter string elements:\n");
for(i=0;i<m;i++)
{
   printf("[%d]=",i );
   scanf("%d",&r[i]);
 }
}
void display()
{
int i,j;
for(i=0;i<m;i++)
     printf("%3d",r[i]);
  printf("\n");
for(i=0;i<n;i++)
 {
     for(j=0;j<m;j++)
     {
        if(me[i][j])
           printf("%3d",me[i][j]);
        else
           printf("       ");
     }
     printf("\n");
 }
  printf("total page faults:%d",f);
}
int search(int p)
{
int i;
for(i=0;i<n;i++)
   if(fr[i]==p)
```

Edit with WPS Office

```
        return 1;
return -1;
}
int get()
{
int i,mi,min=9999;
for(i=0;i<n;i++)
   if(t[i]<min)
   {
        min=t[i];
        mi=i;
    }
 return mi;
}
void lru()
{
int i,j,k;
for(i=0;i<m && s<n;i++)
{
   k=search(r[i]);
   if(k==-1)
   {
      fr[s]=r[i];
      t[s]=i;
      f++;
      s++;
      for(j=0;j<n;j++)
         me[j][i]=fr[j];
    }
   else
      t[k]=i;
 }
 for(i=0;i<m;i++)
 {
   k=search(r[i]);
   if(k==-1)
   {
      s=get();
      fr[s]=r[i];
      t[s]=i;
      f++;
      for(j=0;j<n;j++)
         me[j][i]=fr[j];
    }
     else
        t[k]=i;
   }
   }
   int main()
   {
   accept();
   lru();
   display();
   return 0;
   }
```

Edit with WPS Office

SLIP 12 Question 2

```
#include <stdio.h>
#include <stdlib.h>
int count(char *fname);
int print(char *fname);
int top(char *fname, int c);
int bottom(char *fname, int c, int total);
int total, c, current;

void main(int argc, char *argv[])
{
   char *p;
   if (argc != 4)
      printf("invalid number of arguments");
   if (*argv[2] == '+' || *argv[2] == '-')
   {
      total = count(argv[3]);
      p = argv[2];
      p++;
      c = atoi(p);
      if (c > total)
         printf("invalid line count\n");
      if (*argv[2] == '+')
         top(argv[3], c);
      else if (*argv[2] == '-')
         bottom(argv[3], c, total);
      else
         print(argv[3]);
   }
}

int count(char *fname)
{
   int total, ch;
   FILE *fp;
   fp = fopen(fname, "r");
   if (fp == NULL)
   {
      printf("\nunable to open file");
      return (-1);
   }
   total = 0;
   ch = fgetc(fp);
   while (ch != EOF)
   {
      if (ch == 10)
         total++;
      ch = fgetc(fp);
   }
   total++;
   fclose(fp);
   return (total);
}

int print(char *fname)
{
```

Edit with WPS Office

```
    int ch;
    FILE *fp;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}

int top(char *fname, int c)
{
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (current < c)
    {
        printf("%c", ch);
        if (ch == 10)
            current++;
        ch = fgetc(fp);
    }
    fclose(fp);
}

int bottom(char *fname, int c, int total)
{
    int current;
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (1)
    {
        if (ch == 10)
            current++;
        if (current >= total - c)
            break;
        ch = fgetc(fp);
    }
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}
```

SLIP 14


Edit with WPS Office

(Question.1)

```
#include <stdio.h>
#include <stdlib.h>
int count(char *fname);
int print(char *fname);
int top(char *fname, int c);
int bottom(char *fname, int c, int total);
int total, c, current;

void main(int argc, char *argv[])
{
   char *p;
   if (argc != 4)
      printf("invalid number of arguments");
   if (*argv[2] == '+' || *argv[2] == '-')
   {
      total = count(argv[3]);
      p = argv[2];
      p++;
      c = atoi(p);
      if (c > total)
         printf("invalid line count\n");
      if (*argv[2] == '+')
         top(argv[3], c);
      else if (*argv[2] == '-')
         bottom(argv[3], c, total);
      else
         print(argv[3]);
   }
}

int count(char *fname)
{
   int total, ch;
   FILE *fp;
   fp = fopen(fname, "r");
   if (fp == NULL)
   {
      printf("\nunable to open file");
      return (-1);
   }
   total = 0;
   ch = fgetc(fp);
   while (ch != EOF)
   {
      if (ch == 10)
         total++;
      ch = fgetc(fp);
   }
   total++;
   fclose(fp);
   return (total);
}

int print(char *fname)
{
   int ch;
```

Edit with WPS Office

```
    FILE *fp;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}

int top(char *fname, int c)
{
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (current < c)
    {
        printf("%c", ch);
        if (ch == 10)
            current++;
        ch = fgetc(fp);
    }
    fclose(fp);
}

int bottom(char *fname, int c, int total)
{
    int current;
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (1)
    {
        if (ch == 10)
            current++;
        if (current >= total - c)
            break;
        ch = fgetc(fp);
    }
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}
```

SLIP 14

Question 2

```
#include<stdio.h>
int main() {
```

Edit with WPS Office

```
    int time, burst_time[10], at[10], sum_burst_time = 0, smallest, n, i;
    int sumt = 0, sumw = 0;
    printf("enter the no of processes : ");
    scanf("%d", & n);
    for (i = 0; i < n; i++) {
        printf("the arrival time for process P%d : ", i + 1);
        scanf("%d", & at[i]);
        printf("the burst time for process P%d : ", i + 1);
        scanf("%d", & burst_time[i]);
        sum_burst_time += burst_time[i];
    }
    burst_time[9] = 9999;
    for (time = 0; time < sum_burst_time;) {
        smallest = 9;
        for (i = 0; i < n; i++) {
            if (at[i] <= time && burst_time[i] > 0 && burst_time[i] < burst_time[smallest])
                smallest = i;
        }
        printf("P[%d]\t|\t%d\t|\t%d\n", smallest + 1, time + burst_time[smallest] - at[smallest], time
- at[smallest]);
        sumt += time + burst_time[smallest] - at[smallest];
        sumw += time - at[smallest];
        time += burst_time[smallest];
        burst_time[smallest] = 0;
    }
    printf("\n\n average waiting time = %f", sumw * 1.0 / n);
    printf("\n\n average turnaround time = %f", sumt * 1.0 / n);
    return 0;
}
```

SLIP 16
Question 1

```
#include <stdio.h>
#include <stdlib.h>
int count(char *fname);
int print(char *fname);
int top(char *fname, int c);
int bottom(char *fname, int c, int total);
int total, c, current;

void main(int argc, char *argv[])
{
    char *p;
    if (argc != 4)
        printf("invalid number of arguments");
    if (*argv[2] == '+' || *argv[2] == '-')
    {
        total = count(argv[3]);
        p = argv[2];
        p++;
        c = atoi(p);
        if (c > total)
            printf("invalid line count\n");
        if (*argv[2] == '+')
            top(argv[3], c);
        else if (*argv[2] == '-')
```

Edit with WPS Office

```
            bottom(argv[3], c, total);
        else
            print(argv[3]);
    }
}

int count(char *fname)
{
    int total, ch;
    FILE *fp;
    fp = fopen(fname, "r");
    if (fp == NULL)
    {
        printf("\nunable to open file");
        return (-1);
    }
    total = 0;
    ch = fgetc(fp);
    while (ch != EOF)
    {
        if (ch == 10)
            total++;
        ch = fgetc(fp);
    }
    total++;
    fclose(fp);
    return (total);
}

int print(char *fname)
{
    int ch;
    FILE *fp;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}

int top(char *fname, int c)
{
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (current < c)
    {
        printf("%c", ch);
        if (ch == 10)
            current++;
        ch = fgetc(fp);
    }
```

Edit with WPS Office

```
    fclose(fp);
}

int bottom(char *fname, int c, int total)
{
    int current;
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (1)
    {
        if (ch == 10)
            current++;
        if (current >= total - c)
            break;
        ch = fgetc(fp);
    }
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}
```

SLIP 16 Question 2

```
#include<stdio.h>
int main() {
    int time, burst_time[10], at[10], sum_burst_time = 0, smallest, n, i;
    int sumt = 0, sumw = 0;
    printf("enter the no of processes : ");
    scanf("%d", & n);
    for (i = 0; i < n; i++) {
        printf("the arrival time for process P%d : ", i + 1);
        scanf("%d", & at[i]);
        printf("the burst time for process P%d : ", i + 1);
        scanf("%d", & burst_time[i]);
        sum_burst_time += burst_time[i];
    }
    burst_time[9] = 9999;
    for (time = 0; time < sum_burst_time;) {
        smallest = 9;
        for (i = 0; i < n; i++) {
            if (at[i] <= time && burst_time[i] > 0 && burst_time[i] < burst_time[smallest])
                smallest = i;
        }
        printf("P[%d]\t|\t%d\t|\t%d\n", smallest + 1, time + burst_time[smallest] - at[smallest], time
- at[smallest]);
        sumt += time + burst_time[smallest] - at[smallest];
        sumw += time - at[smallest];
        time += burst_time[smallest];
        burst_time[smallest] = 0;
```

Edit with WPS Office

```
    }
    printf("\n\n average waiting time = %f", sumw * 1.0 / n);
    printf("\n\n average turnaround time = %f", sumt * 1.0 / n);
    return 0;
}
```

SLIP 17
Question 1

```
#include<stdio.h>
int n,page[20],f,fr[20],i;
void display()
{
for(i=0;i<f;i++)
{
printf("%d",fr[i]);
}
printf("\n");
}
void request()
{
                printf("enter no.of pages:");
                scanf("%d",&n);
                printf("enter no of frames:");
                scanf("%d",&f);
                printf("enter pages");
                        for(i=0;i<n;i++)
                        {
                        scanf("%d",&page[i]);
                        }
                        for(i=0;i<n;i++);
                        {
                        fr[i]=-1;
                        }
}
void replace()
{
        int j,flag=0,pf=0;
        int max,lp[10],index,m;
        for(j=0;j<f;j++)
        {
                fr[j]=page[j];
                flag=1;
                pf++;
                display();
        }
        for(j=f;j<n;j++)
        {
                flag=0;
                for(i=0;i<f;i++)
                {
                        if(fr[i]==page[j])
                        {
                        flag=1;
                        break;
                        }
                }
```

Edit with WPS Office

```
                        if(flag==0)
                        {
                                for(i=0;i<f;i++)
                                lp[i]=0;
                                for(i=0;i<f;i++)
                                {
                                for(m=j+1;m<n;m++)
                                        {
                                        if(fr[i]==page[m])
                                            {
lp[i]=m=j;

break;

                                            }
                                            }
                        }
                        max=lp[0];
                        index=0;
                        for(i=0;i<f;i++)
                        {
                              if(lp[i]==0)
                              {
                              index=i;
                              break;
                              }
                              else
                              {
                              if(max<lp[i])
                              {
                              max=lp[i];
                              index=i;
                              }
                              }
                        }
                        fr[index]=page[j];
                pf++;
                display();
                }
}
printf("page faults:%d",pf);
}
void main()
{

request();
replace();
}
```

SLIP 17 Question 2

```
#include<stdio.h>
void main()
{
int i,j,temp,btemp,at[6],bt[6],wt[6],tt[6],sum=0,num,proc[6];
float avrg,avg;
printf("Enter number of processes");
```

Edit with WPS Office

```
scanf("%d",&num);
for(i=0;i<num;i++)
{
printf("\nEnter the process number");
scanf("%d",&proc[i]);
printf("\nEnter the process Arrival time");
scanf("%d",&at[i]);
printf("\nEnter the process Burst time");
scanf("%d",&bt[i]);
}
for(i=0;i<num;i++)
{
for(j=i+1;j<num;j++)
{
if(at[i]>at[j])
{
temp=at[i];
at[i]=at[j];
at[j]=temp;
btemp=bt[i];
bt[i]=bt[j];
bt[j]=btemp;
temp=proc[i];
proc[i]=proc[j];
proc[j]=temp;
}
}
}
printf("\nAfter Sorting on Arrival Time\n");
printf("Process\t Arrival Time\tBurst Time\n");
for(i=0;i<num;i++)
{
printf("\nP%d\t",proc[i]);
printf("%d\t",at[i]);
printf("%d\t",bt[i]);
}
wt[0]=0;
for(i=0;i<num;i++)
{
wt[i+1]=wt[i]+bt[i];
sum+=(wt[i]-at[i]);
}
wt[num]=wt[num-1]+bt[num-1];
printf("\nthe waiting time is \t%d",sum);
avrg=(float)sum/(float)num;
printf("\nthe Average waiting time is\t%f",avrg);
sum=0;
for(i=0;i<num;i++)
{
tt[i]=wt[i+1]-at[i];
sum+=tt[i];
}
printf("\nthe sum of Turnaround time is \t%d",sum);
avg=(float)sum/(float)num;
printf("\nthe Average Turnaround time is \t%f",avg);
printf("\nGNATT CHART\n");
printf("\n----------------------------------\n");
```

Edit with WPS Office

```
for(i=0;i<num;i++)
{
printf("\t|p%d\t",proc[i]);
}
printf("\n-----------------------------------\n");
for(i=0;i<=num;i++)
{
printf("%d\t\t",wt[i]);
}
}
```

SLIP 18

Question 1

```
#include<stdio.h>
#define MAX 20
int fr[MAX],r[MAX],me[MAX][MAX],f,s,m,n,t[MAX];
void accept()
{
int i;
printf("enter no of frames:");
scanf("%d",&n);
printf("enter length of reference string:");
scanf("%d",&m);
printf("enter string elements:\n");
for(i=0;i<m;i++)
{
   printf("[%d]=",i );
   scanf("%d",&r[i]);
 }
}
void display()
{
int i,j;
for(i=0;i<m;i++)
     printf("%3d",r[i]);
  printf("\n");
for(i=0;i<n;i++)
 {
    for(j=0;j<m;j++)
    {
       if(me[i][j])
          printf("%3d",me[i][j]);
       else
          printf("       ");
    }
    printf("\n");
 }
 printf("total page faults:%d",f);
}
int search(int p)
{
int i;
for(i=0;i<n;i++)
   if(fr[i]==p)
      return 1;
```

Edit with WPS Office

```
return -1;
}
int get()
{
int i,mi,min=9999;
for(i=0;i<n;i++)
   if(t[i]<min)
   {
        min=t[i];
        mi=i;
    }
 return mi;
}
void lru()
{
int i,j,k;
for(i=0;i<m && s<n;i++)
{
   k=search(r[i]);
   if(k==-1)
   {
      fr[s]=r[i];
      t[s]=i;
      f++;
      s++;
      for(j=0;j<n;j++)
         me[j][i]=fr[j];
    }
   else
      t[k]=i;
 }
 for(i=0;i<m;i++)
 {
   k=search(r[i]);
   if(k==-1)
   {
      s=get();
      fr[s]=r[i];
      t[s]=i;
      f++;
      for(j=0;j<n;j++)
         me[j][i]=fr[j];
    }
    else
      t[k]=i;
   }
   }
   int main()
   {
   accept();
   lru();
   display();
   return 0;
   }
```

SLIP 18

Edit with WPS Office

Question 2

```
#include<stdio.h>
void main()
{
   int i,j,temp,btemp,at[6],bt[6],wt[6],tt[6],sum=0,num,proc[6];
   float avrg,avg;
   printf("Enter the Number of Processes :");
   scanf("%d",&num);
   for(i=0;i<num;i++)
   {
        printf("\nEnter the Process Number:");
        scanf("%d",&proc[i]);
        printf("\nEnter the Process Arrival Time :");
        scanf("%d",&at[i]);
        printf("\nEnter the Process Burt Time:");
        scanf("%d",&bt[i]);
   }
   for(i=0;i<num;i++)
   {
      for(j=i+1;j<num;j++)
      {
         if(at[i]>at[j])
         {
              temp=at[i];
              at[i]=at[j];
              at[j]=temp;
              btemp=bt[i];
              bt[i]=bt[j];
              bt[j]=btemp;
              temp=proc[i];
              proc[i]=proc[j];
              proc[j]=temp;
         }
      }
   }
   printf("\nAfter Sorting on Arrival Time\n");
   printf("Process\t Arrival Time\tBurst Time\n");
   for(i=0;i<num;i++)
   {
      printf("\nP%d\t",proc[i]);
      printf("%d\t",at[i]);
      printf("%d\t",bt[i]);
   }
   wt[0]=0;
   for(i=0;i<num;i++)
   {
        wt[i+1]=wt[i]+bt[i];
        sum+=(wt[i]-at[i]);
   }
   wt[num]=wt[num-1]+bt[num-1];
   printf("\nThe waiting time is : \t%d",sum);
   avrg=(float)sum/(float)num;
   printf("\nThe average waiting time is :\t%f" ,avrg);
   sum=0;
   for(i=0;i<num;i++)
   {
```

Edit with WPS Office

```
        tt[i]=wt[i+1]-at[i];
        sum+=tt[i];
    }
    printf("\n the sum of Turnaround time is : \t%d",sum);
    avg=(float)sum/(float)num;
    printf("\n the Average Turnaround time is : \t%f",avg);
    printf("\nGantt Chart\n");
    printf("\n.....\n");
    for(i=0;i<num;i++)
    {
         printf("\t|p%d\t",proc[i]);
    }
    printf("\n.....\n");
    for(i=0;i<=num;i++)
    {
        printf("%d\t\t",wt[i]);
    }
  }
```

SLIP 20
Question 1

```
#include <stdio.h>
#include <stdlib.h>
int count(char *fname);
int print(char *fname);
int top(char *fname, int c);
int bottom(char *fname, int c, int total);
int total, c, current;

void main(int argc, char *argv[])
{
    char *p;
    if (argc != 4)
        printf("invalid number of arguments");
    if (*argv[2] == '+' || *argv[2] == '-')
    {
        total = count(argv[3]);
        p = argv[2];
        p++;
        c = atoi(p);
        if (c > total)
            printf("invalid line count\n");
        if (*argv[2] == '+')
            top(argv[3], c);
        else if (*argv[2] == '-')
            bottom(argv[3], c, total);
        else
            print(argv[3]);
    }
}

int count(char *fname)
{
    int total, ch;
    FILE *fp;
    fp = fopen(fname, "r");
```

Edit with WPS Office

```
    if (fp == NULL)
    {
        printf("\nunable to open file");
        return (-1);
    }
    total = 0;
    ch = fgetc(fp);
    while (ch != EOF)
    {
        if (ch == 10)
            total++;
        ch = fgetc(fp);
    }
    total++;
    fclose(fp);
    return (total);
}

int print(char *fname)
{
    int ch;
    FILE *fp;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}

int top(char *fname, int c)
{
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (current < c)
    {
        printf("%c", ch);
        if (ch == 10)
            current++;
        ch = fgetc(fp);
    }
    fclose(fp);
}

int bottom(char *fname, int c, int total)
{
    int current;
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
```

Edit with WPS Office

```
    while (1)
    {
        if (ch == 10)
            current++;
        if (current >= total - c)
            break;
        ch = fgetc(fp);
    }
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}
```

SLIP 20 Question 2

```
#include<stdio.h>
void main()
{
    int i,j,temp,btemp,at[6],bt[6],wt[6],tt[6],sum=0,n,p[5],proc[6];
    float avrg,avg;
    printf("Enter the number of process:");
    scanf("%d",&n);
    for(i=0;i<n;i++)
    {
        printf("\n enter the process number:");
        scanf("%d",&proc[i]);
        printf("\n enter the process arrival time:");
        scanf("%d",&at[i]);
        printf("\n enter the process burst time:");
        scanf("%d",&bt[i]);
        printf("\n enter the burst priority:");
        scanf("%d",&p[i]);
    }
        for(i=0;i<n;i++)
        {
            for(j=i+1;j<n;j++)
            {
                if(at[i]>at[j])
                {
                    temp=at[i];
                    at[i]=at[j];
                    at[j]=temp;
                    btemp=bt[i];
                    bt[i]=bt[j];
                    bt[j]=btemp;
                    temp=proc[i];
                    proc[i]=proc[j];
                    proc[j]=temp;
                }
            }
        }
        for(i=1;i<n;i++)
        {
```

Edit with WPS Office

```
    for(j=i+1;j<n;j++)
    {
       if(p[i]>p[j])
          {
             temp=at[i];
             at[i]=at[j];
             at[j]=temp;
             btemp=bt[i];
             bt[i]=bt[j];
             bt[j]=btemp;
             temp=proc[i];
             proc[i]=proc[j];
             proc[j]=temp;
          }
     }
 }
 for(i=1;i<n;i++)
 {
       if(bt[i]==bt[j])
          {
             if(at[i]>at[j])
             {
                temp=at[i];
                at[i]=at[j];
                at[j]=temp;
                btemp=bt[i];
                bt[i]=bt[j];
                bt[j]=btemp;
                temp=proc[i];
                proc[i]=proc[j];
                proc[j]=temp;
          }
       }
  }
printf("\n After Sorting on arrival time\n");
printf("Process\t Arrival time\t Burst time\t priority\n");
for(i=0;i<n;i++)
{
    printf("\nP%d\t",proc[i]);
    printf("%d\t",at[i]);
    printf("%d\t",bt[i]);
    printf("%d\t",p[i]);
}
   wt[0]=0;
   for(i=0;i<n;i++)
   {
      wt[i+1]=wt[i]+bt[i];
      sum += (wt[i]-at[i]);
   }
   wt[n]=wt[n-1]+bt[n-1];
   printf("\nthe waiting time is \t%d",sum);
   avrg=(float)sum/(float)n;
   printf("\nthe average waiting time is \t%f",avrg);
   sum=0;
   for(i=0;i<n;i++)
   {
    tt[i]=wt[i+1]-at[i];
```

Edit with WPS Office

```
        sum+=tt[i];
      }
      printf("\n the sum of turnaround time is \t%d",sum);
      avg=(float)sum/(float)n;
      printf("\n the average turnaround time is \t%f",avg);
      printf("\n Gantt chart\n");
      printf("\n-------------------------------------------------\n");
      for(i=0;i<n;i++)
      {
         printf("\t|p%d\t",proc[i]);
      }
         printf("\n-------------------------------------------------\n");
         for(i=0;i<=n;i++)
         {
         printf("%d\t\t\t",wt[i]);
      }
  }
```

SLIP 22
Question 1

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>

int main()
{
      // fork() Create a child process

      int pid = fork();
      if (pid > 0) {
            printf("I am Parent process\n");
            printf("ID : %d\n\n", getpid());
      }
      else if (pid == 0) {
            printf("I am Child process\n");
            // getpid() will return process id of child process
            printf("ID: %d\n", getpid());

      }
      else {
            printf("Failed to create child process");
      }

      return 0;
}
```

SLIP 22
Question 2

```
#include<stdio.h>
void main()
{
   int i,j,temp,btemp,at[6],bt[6],wt[6],tt[6],sum=0,n,p[5],proc[6];
   float avrg,avg;
   printf("Enter the number of process:");
   scanf("%d",&n);
```

Edit with WPS Office

```
for(i=0;i<n;i++)
{
   printf("\n enter the process number:");
   scanf("%d",&proc[i]);
   printf("\n enter the process arrival time:");
   scanf("%d",&at[i]);
   printf("\n enter the process burst time:");
   scanf("%d",&bt[i]);
   printf("\n enter the burst priority:");
   scanf("%d",&p[i]);
}
   for(i=0;i<n;i++)
   {
      for(j=i+1;j<n;j++)
      {
         if(at[i]>at[j])
         {
            temp=at[i];
            at[i]=at[j];
            at[j]=temp;
            btemp=bt[i];
            bt[i]=bt[j];
            bt[j]=btemp;
            temp=proc[i];
            proc[i]=proc[j];
            proc[j]=temp;
         }
      }
   }
   for(i=1;i<n;i++)
   {
      for(j=i+1;j<n;j++)
      {
         if(p[i]>p[j])
            {
               temp=at[i];
               at[i]=at[j];
               at[j]=temp;
               btemp=bt[i];
               bt[i]=bt[j];
               bt[j]=btemp;
               temp=proc[i];
               proc[i]=proc[j];
               proc[j]=temp;
            }
       }
   }
   for(i=1;i<n;i++)
   {
         if(bt[i]==bt[j])
            {
               if(at[i]>at[j])
               {
                  temp=at[i];
                  at[i]=at[j];
                  at[j]=temp;
                  btemp=bt[i];
```

Edit with WPS Office

```
                bt[i]=bt[j];
                bt[j]=btemp;
                temp=proc[i];
                proc[i]=proc[j];
                proc[j]=temp;
            }
        }
    }
  printf("\n After Sorting on arrival time\n");
  printf("Process\t Arrival time\t Burst time\t priority\n");
  for(i=0;i<n;i++)
  {
     printf("\nP%d\t",proc[i]);
     printf("%d\t",at[i]);
     printf("%d\t",bt[i]);
     printf("%d\t",p[i]);
  }
     wt[0]=0;
     for(i=0;i<n;i++)
     {
        wt[i+1]=wt[i]+bt[i];
        sum += (wt[i]-at[i]);
     }
     wt[n]=wt[n-1]+bt[n-1];
     printf("\nthe waiting time is \t%d",sum);
     avrg=(float)sum/(float)n;
     printf("\nthe average waiting time is \t%f",avrg);
     sum=0;
     for(i=0;i<n;i++)
     {
      tt[i]=wt[i+1]-at[i];
      sum+=tt[i];
     }
     printf("\n the sum of turnaround time is \t%d",sum);
     avg=(float)sum/(float)n;
     printf("\n the average turnaround time is \t%f",avg);
     printf("\n Gantt chart\n");
     printf("\n-------------------------------------------------\n");
     for(i=0;i<n;i++)
     {
        printf("\t|p%d\t",proc[i]);
     }
        printf("\n-------------------------------------------------\n");
        for(i=0;i<=n;i++)
        {
        printf("%d\t\t\t",wt[i]);
     }
  }
```

SLIP 23
Question 1

```
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
int main()
{
```

Edit with WPS Office

```
 int pid;
 pid=getpid();
 printf("Current Process ID is : %d\n",pid);
 printf("\n[Forking Child Process ... ] \n");
 pid=fork();
 if(pid < 0)
 {
  printf("\nProcess can not be created ");
 }
 else
 {
            if(pid==0)
  {
   printf("\nChild Process is Sleeping ...");
   sleep(5);
   printf("\nOrphan Child's Parent ID : %d",getppid());
  }

  else
  { /* Parent Process */
   printf("\nParent Process Completed ...");
  }
 }
 return 0;
}
```

SLIP 23 Question 2

```
#include<stdio.h>
int n,page[20],f,fr[20],i;
void display()
{
for(i=0;i<f;i++)
{
printf("%d",fr[i]);
}
printf("\n");
}
void request()
{
                printf("enter no.of pages:");
                scanf("%d",&n);
                printf("enter no of frames:");
                scanf("%d",&f);
                printf("enter pages");
                        for(i=0;i<n;i++)
                        {
                        scanf("%d",&page[i]);
                        }
                        for(i=0;i<n;i++);
                        {
                        fr[i]=-1;
                        }
}
void replace()
{
        int j,flag=0,pf=0;
```

Edit with WPS Office

```
int max,lp[10],index,m;
for(j=0;j<f;j++)
{
        fr[j]=page[j];
        flag=1;
        pf++;
        display();
}
for(j=f;j<n;j++)
{
        flag=0;
        for(i=0;i<f;i++)
        {
                if(fr[i]==page[j])
                {
                flag=1;
                break;
                }
        }
        if(flag==0)
        {
                for(i=0;i<f;i++)
                lp[i]=0;
                for(i=0;i<f;i++)
                {
                for(m=j+1;m<n;m++)
                        {
                        if(fr[i]==page[m])
                            {
lp[i]=m=j;

break;

                                        }
                                        }
                }
                max=lp[0];
                index=0;
                for(i=0;i<f;i++)
                {
                    if(lp[i]==0)
                    {
                    index=i;
                    break;
                    }
                    else
                    {
                    if(max<lp[i])
                    {
                    max=lp[i];
                    index=i;
                    }
                    }
                }
                fr[index]=page[j];
        pf++;
        display();
```

Edit with WPS Office

```
                        }
}
printf("page faults:%d",pf);
}
void main()
{

request();
replace();

}
```

SLIP 24
Question 1

```
#include<stdio.h>
#include<sys/types.h>
#include<unistd.h>
#include<stdlib.h>

void bubblesort(int arr[30],int n)
{
    int i,j,temp;
    for(i=0;i<n;i++)
    {
        for(j=0;j<n-1;j++)
        {
            if(arr[j]>arr[j+1])
            {
                temp=arr[j];
                arr[j]=arr[j+1];
                arr[j+1]=temp;
            }
        }
    }
}

void insertionsort(int arr[30], int n)
{
    int i, j, temp;
    for (i = 1; i < n; i++) {
        temp = arr[i];
        j = i - 1;

        while(j>=0 && temp <= arr[j])
        {
            arr[j+1] = arr[j];
            j = j-1;
        }
        arr[j+1] = temp;
    }
}
void fork1()
{
    int arr[25],arr1[25],n,i,status;
    printf("\nEnter the no of values in array :");
    scanf("%d",&n);
```

Edit with WPS Office

```
        printf("\nEnter the array elements :");
        for(i=0;i<n;i++)
                scanf("%d",&arr[i]);
        int pid=fork();
        if(pid==0)
        {
                sleep(10);
                printf("\nchild process\n");
                printf("child process id=%d\n",getpid());
                insertionsort(arr,n);
                printf("\nElements Sorted Using insertionsort:");
                printf("\n");
                for(i=0;i<n;i++)
                        printf("%d,",arr[i]);
                printf("\b");
                printf("\nparent process id=%d\n",getppid());
                system("ps -x");
          }
         else
          {
                printf("\nparent process\n");
                printf("\nparent process id=%d\n",getppid());
            bubblesort(arr,n);
            printf("Elements Sorted Using bubblesort:");
                printf("\n");
                for(i=0;i<n;i++)
                        printf("%d,",arr[i]);
                printf("\n\n\n");
         }
 }
 int main()
 {
      fork1();
      return 0;
 }
```

SLIP 24
Question 2

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
void make_toks(char *s, char *tok[])
{
int i=0;
char *p;
p = strtok(s," ");
while(p!=NULL)
{
  tok[i++]=p;
  p=strtok(NULL," ");
}
tok[i]=NULL;
}
```

Edit with WPS Office

```
void count(char *fn, char op)
{
int fh,cc=0,wc=0,lc=0;
char c;
fh = open(fn,O_RDONLY);
if(fh==-1)
{
  printf("File %s not found.\n",fn);
return;
}
while(read(fh,&c,1)>0)
{
if(c==' ') wc++;
else if(c=='\n')
  {
  wc++;
  lc++;
  }
  cc++;
}
close(fh);
switch(op)
{
case 'c':
  printf("No.of characters:%d\n",cc-1);
break;
case 'w':
  printf("No.of words:%d\n",wc);
break;
case 'l':
  printf("No.of lines:%d\n",lc+1);
break;
}
}
int main()
{
char buff[80],*args[10];
int pid;
while(1)
{
  printf("myshell$ ");
  fflush(stdin);
  fgets(buff,80,stdin);
  buff[strlen(buff)-1]='\0';
  make_toks(buff,args);
if(strcmp(args[0],"count")==0)
  count(args[2],args[1][0]);
else
  {
  pid = fork();
  if(pid>0)
  wait();
  else
  {
  if(execvp(args[0],args)==-1)
  printf("Bad command.\n");
  }
```

Edit with WPS Office

```
 }
}
return 0;
}
```

SLIP 25
Question 2

```
#include <stdio.h>
#include <stdlib.h>
int count(char *fname);
int print(char *fname);
int top(char *fname, int c);
int bottom(char *fname, int c, int total);
int total, c, current;

void main(int argc, char *argv[])
{
   char *p;
   if (argc != 4)
      printf("invalid number of arguments");
   if (*argv[2] == '+' || *argv[2] == '-')
   {
      total = count(argv[3]);
      p = argv[2];
      p++;
      c = atoi(p);
      if (c > total)
         printf("invalid line count\n");
      if (*argv[2] == '+')
         top(argv[3], c);
      else if (*argv[2] == '-')
         bottom(argv[3], c, total);
      else
         print(argv[3]);
   }
}

int count(char *fname)
{
   int total, ch;
   FILE *fp;
   fp = fopen(fname, "r");
   if (fp == NULL)
   {
      printf("\nunable to open file");
      return (-1);
   }
   total = 0;
   ch = fgetc(fp);
   while (ch != EOF)
   {
      if (ch == 10)
         total++;
      ch = fgetc(fp);
   }
   total++;
```

Edit with WPS Office

```
    fclose(fp);
    return (total);
}

int print(char *fname)
{
    int ch;
    FILE *fp;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
    fclose(fp);
}

int top(char *fname, int c)
{
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (current < c)
    {
        printf("%c", ch);
        if (ch == 10)
            current++;
        ch = fgetc(fp);
    }
    fclose(fp);
}

int bottom(char *fname, int c, int total)
{
    int current;
    int ch;
    FILE *fp;
    current = 0;
    fp = fopen(fname, "r");
    ch = fgetc(fp);
    while (1)
    {
        if (ch == 10)
            current++;
        if (current >= total - c)
            break;
        ch = fgetc(fp);
    }
    ch = fgetc(fp);
    while (ch != EOF)
    {
        printf("%c", ch);
        ch = fgetc(fp);
    }
```


Edit with WPS Office

```
    fclose(fp);
}
```

Edit with WPS Office