$printf("\n\%d\t\%d\t\t\%d\t\t\%d",i$ 

<pre>#include<stdio.h> #include<conio.h> #include<conio.h> #define max 30 void main() {     Int     i,j,n,bt[max],at[max],wt[max],tat[     max],temp[max];     float awt=0,atat=0;     printf("Enter the number of     process:");     scanf("%d",&amp;n);     printf("Enter the burst time:");     for(i=0;i<n;i++) (i="0;" +="" 1);="" <="" arrival="" atat="atat+tat[i];" awt="awt+wt[i];" chart:\n");="" for="" for(i="0;i&lt;n;i++)" i="" i++)="" n;="" p%d",="" pre="" printf("\ngantt="" printf("enter="" printf(" ="" printf(" rocess\tburst="" scanf("%d",&at[i]);="" scanf("%d",&bt[i]);="" tat[i]="wt[i]+bt[i];" temp[0]="0;" temp[i+1]="temp[i]+bt[i];" the="" time:");="" time\n");="" time\tarrival="" time\tturnaround="" time\twaiting="" wt[i]="temp[i]-at[i];" {="" }="" }<=""></n;i++)></conio.h></conio.h></stdio.h></pre>	<pre>#include<stdio.h> #include<conio.h> #include<stdlib.h> void main() {     int     i,n,qt,count=0,temp,sq=0,bt[10],     wt[10],tat[10],rem_bt[10];     float awt=0,atat=0;     printf("Enter the number of process:");     scanf("%d",&amp;n);     printf("Enter the burst time:");     for(i=0;i<n;i++) continue;="" count++;="" if(rem_bt[i]="" printf("enter="" quantum="" rem_bt[i]="bt[i];" scanf("%d",&bt[i]);="" scanf("%d",&qt);="" temp="qt;" the="" time:");="" while(1)="" {="" }="">qt)             rem_bt[i]-qt;         else         if(rem_bt[i]&gt;=0)         {             temp=rem_bt[i];             rem_bt[i]=o;         }         sq=sq+temp;         tat[i]=sq;     } }</n;i++)></stdlib.h></conio.h></stdio.h></pre>	+1,bt[i],tat[i],wt[i]); } awt=awt/n; atat=atat/n; printf("\nAverage waiting time=%f",awt); printf("\nAverage turn around time=%f",atat); }  SJF  #include <stdio.h> #include<stdlib.h> #include<conio.h> #define max 30 void main() {     int     i,j,n,temp,p[max],at[max],bt[max ],wt[max],tat[max];     float awt=0,atat=0;     printf("Enter the number of process:");     scanf("%d",&amp;n);     printf("Enter the process number: ");     for(i=0;i<n;i++) arrival="" burst="" each="" for="" for(i="0;i&lt;n;i++)" printf("enter="" process:");="" scanf("%d",&at[i]);="" scanf("%d",&bt[i]);="" scanf("%d",&p[i]);="" th="" the="" time="" {="" }="" }<=""></n;i++)></conio.h></stdlib.h></stdio.h>
printf("%d\t\t%d\t\t%d\t\t	if(n==count) break;	} for(i=0;i <n;i++)< td=""></n;i++)<>
%d\n",i+1,bt[i],at[i],wt[i],tat[i]); }	}	{ for(j=0;j <n-i-1;j++)< td=""></n-i-1;j++)<>
awt = awt/n; atat = atat/n;	<pre>printf("\nProcess\tBurstTime\tTu rnAroundTime\tWaitingTime\n");</pre>	ι if(bt[j]>bt[j+1])
<pre>printf("Average Waiting Time =</pre>	for(i=0;i <n;i++)< td=""><td>{</td></n;i++)<>	{
%f\n",awt);	{	temp=bt[j];
<pre>printf("Average Turnaround Time = %f\n",atat);</pre>	wt[i]=tat[i]-bt[i]; awt=awt+wt[i];	bt[j]=bt[j+1]; bt[j+1]=temp;
c - /or (ii ,atat),	awt-awt+wt[i]; atat=atat+tat[i];	temp=at[j];
}	acac acac cacjij	at[j]=at[j+1];
•		at[j+1]=temp;
		- 1,

```
temp=p[j];
                                                                                          for (i = 0; i < reference_size;</pre>
                                              i = 0;
         p[j]=p[j+1];
                                              printf("\tref string\t page
                                                                                        i++) {
         p[j+1]=temp;
                                            frames\n");
                                                                                             scanf("%d", &reference[i]);
       }
                                              for (i = 1; i <= n; i++)
                                                                                          }
    }
                                              {
                                                                                          for (i = 0; i < n; i++) {
                                                 printf("%d\t\t", ref_string[i]);
  printf("\nGantt Chart:\n");
                                                 avail = 0;
                                                                                             frames[i] = -1; // Initialize
                                                                                        frames with -1 to indicate empty
  for (i = 0; i < n; i++) {
                                                for (k = 0; k < no; k++)
    printf("| P%d ", i + 1);
                                                   if (frame[k] ==
                                                                                        frame
                                            ref_string[i])
                                                                                             counters[i] = 0; // Initialize
                                                                                        counters to 0
  printf("|\n");
                                                     avail = 1;
  printf("\n");
                                                if (avail == 0)
                                                                                          }
  printf("process \t arrival time
\t burst time \t waiting time \t
                                                   frame[j] = ref_string[i];
                                                                                          printf("\nPage Scheduling:\n");
turn around time\n");
                                                   j = (j + 1) \% no;
  for(i=0;i<n;i++)
                                                   count++;
                                                                                          for (i = 0; i < reference_size;
                                                   for (k = 0; k < no; k++)
                                                                                        i++) {
                                                     printf("%d\t", frame[k]);
    wt[i]=0;
                                                                                             int page = reference[i];
    tat[i]=0;
                                                }
                                                                                             int page_found = 0;
    for(j=0;j<i;j++)
                                                 printf("\n");
                                                                                             for (j = 0; j < n; j++) {
                                              }
                                                                                               if (frames[j] == page) {
    {
                                              printf("Page Fault Is %d",
                                                                                                  page_found = 1;
       wt[i]=wt[i]+bt[j];
                                                                                                 counters[j]++;
                                            count);
    tat[i]=wt[i]+bt[i];
                                              return 0;
                                                                                                  break;
    awt=awt+wt[i];
                                                                                               }
    atat=atat+tat[i];
                                                                                             }
    printf("%d\t\t %d\t\t %d\t\t
                                            LFU
                                                                                             if (!page found) {
%d\t\t
                                                                                               int min_count =
%d\n",p[i],at[i],bt[i],wt[i],tat[i]);
                                            #include <stdio.h>
                                                                                        counters[0];
  }
                                            #include <stdlib.h>
                                                                                               int min index = 0;
  awt=awt/n;
                                                                                               for (j = 1; j < n; j++) {
  atat=atat/n;
                                            #define MAX_PAGES 100
                                                                                                  if (counters[j] <
  printf("Average waiting
                                                                                        min_count) {
time=%f \n",awt);
                                            int main() {
                                                                                                    min_count =
  printf("Average turn around
                                              int n, i, j, page_faults = 0;
                                                                                        counters[j];
time=%f \n",atat);
                                                                                                    min index = j;
                                                                                                 }
                                              printf("Enter the number of
                                            frames: ");
                                                                                               }
FIFO
                                              scanf("%d", &n);
                                                                                               frames[min index] = page;
#include <stdio.h>
                                              if (n \le 0)
                                                                                               counters[min index] = 1;
                                                printf("Invalid number of
                                                                                        page_faults++;
int main()
                                                                                               printf("Page %d loaded
                                            frames. Exiting...\n");
{
  int i, j, n, ref_string[50],
                                                 return 1;
                                                                                        into frame %d\n", page,
frame[10], no, k, avail, count = 0;
                                              }
                                                                                        min_index);
  printf("\n ENTER THE NUMBER
                                                                                             }
OF PAGES:\n");
                                              int frames[n], counters[n],
                                                                                          }
  scanf("%d", &n);
                                            reference[MAX_PAGES];
                                                                                          printf("\nTotal Page Faults:
  printf("\n ENTER THE PAGE
NUMBER:\n");
                                              printf("Enter the reference
                                                                                        %d\n", page_faults);
  for (i = 1; i <= n; i++)
                                            string size: ");
    scanf("%d", &ref string[i]);
                                              int reference size;
                                                                                          return 0;
  printf("\n ENTER THE NUMBER
                                              scanf("%d", &reference_size);
                                                                                        }
OF FRAMES:");
  scanf("%d", &no);
                                              printf("Enter the reference
  for (i = 0; i < no; i++)
                                            string (space-separated): ");
```

frame[i] = -1;

```
LRU
                                                            if (b[r] < b[j])
                                                                                              for (j = 0; j < noFrames; j++)
#include <stdio.h>
                                                               t = b[r];
                                                                                                frameFREQ[i] = 0;
                                                               b[r] = b[j];
main()
{
                                                               b[i] = t;
                                                                                              for (j = 0; j < noPages; j++)
  int q[20], p[50], c = 0, c1, d, f, i,
                                                            }
                                                          }
j, k = 0, n, r, t, b[20], c2[20];
                                                                                                int index;
                                                                                                 printf(" page:%d ",
  printf("Enter no of pages:");
  scanf("%d", &n);
                                                        for (r = 0; r < f; r++)
                                                                                            pages[i]);
  printf("Enter the reference
                                                                                                flagFound = 0, flag = 0, flag2
                                                          if (c2[r] == b[0])
string:");
                                                                                           = 0;
  for (i = 0; i < n; i++)
                                                            q[r] = p[i];
    scanf("%d", &p[i]);
                                                          printf("\t^{g}d", q[r]);
                                                                                                for (i = 0; i < noFrames; i++)
  printf("Enter no of frames:");
  scanf("%d", &f);
                                                        printf("\n");
                                                                                                   if (frames[i] == pages[j])
  q[k] = p[k];
                                                     }
  printf("\n\t\%d\n", q[k]);
                                                   }
                                                                                                     flagFound = 1;
                                                                                                      flag = 1;
  C++;
  k++;
                                                printf("\nThe no of page faults
                                                                                                      index = i;
                                              is %d", c);
                                                                                                      printf("hit ");
  for (i = 1; i < n; i++)
    c1 = 0;
                                                                                                      hit++;
    for (j = 0; j < f; j++)
                                              MFU
                                                                                                     break;
       if (p[i] != q[j])
                                              #include <stdio.h>
                                                                                                   }
                                              #include <stdlib.h>
                                                                                                 }
         c1++;
    if (c1 == f)
                                              void main()
                                                                                                 if (flagFound == 0) // if frame
                                                                                            not found and empty frame
                                                int hit = 0, miss = 0, i, j,
                                                                                            avalible
       C++;
       if (k < f)
                                              noPages, noFrames, min;
                                                int frames[10], pages[20];
                                                                                                   for (i = 0; i < noFrames;
                                                int flag = 0, flag1 = 0, flag2 = 0;
         q[k] = p[i];
                                                                                           i++)
                                                int flagFound = 0;
         k++;
         for (j = 0; j < k; j++)
                                                int count = 0:
                                                                                                      if (frames[i] == -1)
            printf("\t%d", q[j]);
                                                int frameAge[50],
         printf("\n");
                                              frameFREQ[50];
                                                                                                        frames[i] = pages[j];
       }
                                                                                                        flag = 1;
                                                printf("enter number of
       else
                                                                                                        count++;
                                              frames\n");
                                                                                                        frameAge[i] = count;
       {
         for (r = 0; r < f; r++)
                                                scanf("%d", &noFrames);
                                                printf("enter number of
                                                                                                        miss++;
            c2[r] = 0;
                                              pages\n");
                                                                                                        frameFREQ[i] = 1;
            for (j = i - 1; j < n; j--)
                                                scanf("%d", &noPages);
                                                                                                        printf("miss");
                                                                                                        break;
              if (q[r] != p[j])
                                                printf("enter the page string ");
                                                                                                     }
                                                                                                   }
                 c2[r]++;
                                                for (i = 0; i < noPages; i++)
              else
                 break;
                                                                                                   if (flag == 0)
            }
                                                   scanf("%d", &pages[i]);
                                                                                                     int bestmfu = 0;
         for (r = 0; r < f; r++)
                                                                                                     for (i = 0; i < noFrames;
                                                for (i = 0; i < noFrames; i++)
            b[r] = c2[r];
                                                                                           i++)
         for (r = 0; r < f; r++)
                                                   frames[i] = -1;
                                                                                                        if (frameFREQ[i] >
            for (j = r; j < f; j++)
                                                   frameAge[i] = -1;
                                                                                            frameFREQ[bestmfu])
                                                                                                          bestmfu = i;
```

```
int m = 0;
         frames[bestmfu] =
                                               frames[i] = -1;
                                                                                           for (i = 0; i < noFrames;
pages[j];
                                               frameAge[i] = -1;
                                                                                     i++)
         miss++;
         printf("miss ");
                                                                                              if (frameAge[i] > max) //
                                                                                     only this line changes for mru
         frameFREQ[bestmfu] =
                                            printf("\nPageNo miss/hit
1;
                                          frames");
                                                                                     compared to Iru
                                            printf("\n-----
      }
                                            -----\n");
                                                                                                max = frameAge[i];
          else
                                            for (j = 0; j < noPages; j++)
    }
                                                                                                m = i;
                                                                                             }
                                                                                           }
      frameFREQ[index]++;
                                               printf("%3d ", pages[j]);
                                               flagFound = 0, flag = 0, flag2
    }
                                                                                           frames[m] = pages[j];
                                          = 0;
    for (i = 0; i < noFrames; i++)
                                                                                           count++;
                                               for (i = 0; i < noFrames; i++)
                                                                                           frameAge[m] = count;
      printf(" %d ", frames[i]);
                                                                                           miss++;
                                                                                           printf("miss\t");
                                                 if (frames[i] == pages[j])
    printf("\n");
                                                                                         }
                                                   flagFound = 1;
                                                   flag = 1;
                                                                                         for (i = 0; i < noFrames; i++)
  printf("number of hits %d\n",
                                                   count++;
                                                                                           printf(" %3d ", frames[i]);
                                                   frameAge[i] = count;
  printf("number of miss %d\n",
                                                   // age frame
miss);
                                                   hit++;
                                                                                         printf("\n");
                                                   printf("hit \t");
                                                   break;
MRU
                                                 }
                                                                                       printf("\n-----
                                                                                         -----\n");
                                               }
#include <stdio.h>
                                                                                       printf("number of hits %d\n",
#include <stdlib.h>
                                               if (flagFound == 0) // if frame
                                          not found and empty frame
                                                                                    hit):
void main()
                                          avalible
                                                                                       printf("number of miss %d\n",
                                                                                     miss);
  int hit = 0, miss = 0, i, j,
                                                 for (i = 0; i < noFrames;
                                                                                     }
noPages, noFrames, max;
                                          i++)
  int frames[10], pages[20];
                                                                                    SLIP 21
  int flag = 0, flag1 = 0, flag2 = 0;
  int flagFound = 0;
                                                   if (frames[i] == -1)
                                                                                    // Write a C Program to create a
  int count = 0;
                                                                                     child process using fork (), display
                                                     frames[i] = pages[j];
  int frameAge[10];
                                                                                     parent and
                                                                                    // child process id. Child process
  printf("enter number of
                                                                                    will display the message "I am
                                                     flag = 1;
frames\n");
                                                     count++;
                                                                                     Child Process"
  scanf("%d", &noFrames);
                                                     frameAge[i] = count;
                                                                                     // and the parent process should
  printf("enter number of
                                                     printf("miss\t");
                                                                                     display "I am Parent Process".
pages\n");
                                                     miss++;
  scanf("%d", &noPages);
                                                                                     #include <stdio.h>
                                                     break;
                                                                                     #include <unistd.h>
                                                   }
  printf("enter the page string ");
                                                                                     #include <sys/types.h>
                                                 }
  for (i = 0; i < noPages; i++)
                                                                                    int main()
                                              if (flag == 0) // if frame not
    scanf("%d", &pages[i]);
                                          found
                                                                                       pid_t pid;
                                                 // printf("check\n");
                                                                                       pid = fork();
  for (i = 0; i < noFrames; i++)
                                                 max = frameAge[0];
```

```
if (pid < 0)
                                                printf("Child process is doing
                                                                                       sleeping for 5 seconds.\n",
                                            some work...\n");
                                                                                       getpid());
    fprintf(stderr, "Fork
                                                sleep(5);
                                                                                            sleep(5);
failed\n");
                                                                                            printf("Child process is done
                                                printf("Child process
                                                                                       with its task.\n");
    return 1;
                                                                                            printf("Child process (PID:
  }
                                            exiting.\n");
  else if (pid == 0)
                                                exit(EXIT_SUCCESS);
                                                                                       %d) is now an orphan process
                                                                                       with PPID: %d.\n", getpid(),
                                              }
    printf("I am Child
                                              else
                                                                                       getppid());
Process\n");
    printf("Child Process ID:
                                                wait(NULL);
                                                                                          }
                                                printf("Parent process ID:
%d\n", getpid());
                                            %d\n", getpid());
                                                                                          return 0;
  }
  else
                                                printf("Parent process
                                                                                       }
                                            exiting.\n");
    printf("I am Parent
                                                exit(EXIT_SUCCESS);
                                                                                       SLIP 24
Process\n");
                                              }
    printf("Parent Process ID:
                                                                                       // Write a C program to accept n
%d\n", getpid());
                                              return 0;
                                                                                       integers to be sorted. Main
                                                                                       function
  }
                                            }
                                                                                       // creates child process using fork
  return 0;
                                           SLIP 23
                                                                                       system call. Parent process sorts
}
                                                                                       the integers
                                           // Write a C program to illustrate
                                                                                       // using bubble sort and waits for
SLIP 22
                                            the concept of orphan process.
                                                                                       child process using wait system
                                            Parent process, creates a child
                                                                                       call. Child
                                            and terminates before child has
#include <stdio.h>
                                                                                       // process sorts the integers
#include <stdlib.h>
                                            finished its task. So child process
                                                                                       using insertion sort.
#include <unistd.h>
                                            becomes orphan process. (Use
                                            fork(), sleep(), getpid(),
                                                                                       #include <stdio.h>
int main()
                                            getppid()).
                                                                                       #include <stdlib.h>
                                                                                       #include <sys/types.h>
                                            #include <stdio.h>
                                                                                       #include <sys/wait.h>
  pid_t pid;
                                            #include <stdlib.h>
                                                                                       #include <unistd.h>
  pid = fork();
                                            #include <unistd.h>
                                                                                       void bubbleSort(int arr[], int n);
                                            #include <sys/types.h>
                                                                                       void insertionSort(int arr[], int n);
  if (pid < 0)
                                            #include <sys/wait.h>
                                                                                       int main() {
    perror("Fork failed");
                                            int main() {
                                                                                          int n;
    exit(EXIT FAILURE);
                                              pid t pid = fork();
                                                                                          printf("Enter the number of
                                              if (pid < 0) {
                                                perror("Fork failed");
                                                                                       integers: ");
  else if (pid == 0)
                                                                                          scanf("%d", &n);
                                                exit(EXIT_FAILURE);
    printf("Child process ID:
                                              }
%d\n", getpid());
                                                                                          int arr[n];
    int nice_val = nice(10); // Set
                                              if (pid > 0) {
a higher priority (lower nice
                                                printf("Parent process (PID:
                                                                                          printf("Enter %d integers:\n",
                                            %d) is sleeping for 2 seconds and
value)
                                                                                       n);
    if (nice_val == -1)
                                            then terminating.\n", getpid());
                                                                                          for (int i = 0; i < n; i++) {
                                                                                            scanf("%d", &arr[i]);
                                                sleep(2);
       perror("Nice failed");
                                                printf("Parent process is
                                                                                          }
       exit(EXIT_FAILURE);
                                            terminating.\n");
    }
                                                exit(EXIT_SUCCESS);
                                                                                          pid t pid = fork();
                                                printf("Child process (PID:
    printf("Child process nice
                                                                                          if (pid < 0) {
value: %d\n", nice_val);
                                            %d) is doing some task and
                                                                                            fprintf(stderr, "Fork
                                                                                       failed\n");
```

```
printf("\n");
     exit(EXIT_FAILURE);
                                                   arr[j + 1] = key;
  } else if (pid == 0) {}
                                                }
                                                                                               } else {
    // Child process
                                              }
                                                                                                 wait(NULL);
     printf("Child process sorting
                                                                                                  printf("Parent process
using Insertion Sort.\n");
                                              SLIP 25
                                                                                            sorting using Bubble Sort.\n");
     insertionSort(arr, n);
                                                                                                 bubbleSort(arr, n);
                                              // Write a C program to accept n
     printf("Sorted array in child
                                                                                                  printf("Sorted array in
process:\n");
                                              integers to be sorted. Main
                                                                                            parent process:\n");
     for (int i = 0; i < n; i++) {
                                              function
                                                                                                 for (int i = 0; i < n; i++) {
       printf("%d ", arr[i]);
                                              // creates child process using fork
                                                                                                    printf("%d ", arr[i]);
                                              system call. Parent process sorts
     }
     printf("\n");
                                              the integers
                                                                                                 printf("\n");
                                              // using bubble sort and waits for
                                                                                               }
  } else {
     wait(NULL);
                                              child process using wait system
     printf("Parent process
                                              call. Child
                                                                                               return 0;
sorting using Bubble Sort.\n");
                                              // process sorts the integers
                                                                                            }
     bubbleSort(arr, n);
                                                                                            void bubbleSort(int arr[], int n) {
                                              using insertion sort.
     printf("Sorted array in
                                                                                               for (int i = 0; i < n - 1; i++) {
parent process:\n");
                                              #include <stdio.h>
                                                                                                 for (int j = 0; j < n - i - 1; j++) {
     for (int i = 0; i < n; i++) {
                                              #include <stdlib.h>
                                                                                                    if (arr[j] > arr[j + 1]) {
       printf("%d ", arr[i]);
                                              #include <sys/types.h>
                                                                                                      int temp = arr[j];
    }
                                              #include <sys/wait.h>
                                                                                                      arr[i] = arr[i + 1];
                                              #include <unistd.h>
     printf("\n");
                                                                                                      arr[j + 1] = temp;
  }
                                                                                                    }
                                              void bubbleSort(int arr[], int n);
                                                                                                 }
  return 0;
                                              void insertionSort(int arr[], int n);
                                                                                               }
}
                                                                                            }
                                                                                            void insertionSort(int arr[], int n)
Sort
                                              int main() {
void bubbleSort(int arr[], int n) {
                                                int n;
  for (int i = 0; i < n - 1; i++) {
                                                                                               int i, key, j;
     for (int j = 0; j < n - i - 1; j++) {
                                                printf("Enter the number of
                                                                                               for (i = 1; i < n; i++) {
       if (arr[j] > arr[j + 1]) {
                                              integers: ");
                                                                                                 key = arr[i];
         // Swap arr[j] and
                                                scanf("%d", &n);
                                                                                                 j = i - 1;
arr[j+1]
                                                                                                 while (j \ge 0 \&\& arr[j] > key)
         int temp = arr[j];
                                                int arr[n];
                                                                                            {
         arr[j] = arr[j + 1];
                                                                                                    arr[j + 1] = arr[j];
         arr[j + 1] = temp;
                                                printf("Enter %d integers:\n",
                                                                                                    j = j - 1;
       }
                                              n);
    }
                                                for (int i = 0; i < n; i++) {
                                                                                                 arr[j + 1] = key;
                                                   scanf("%d", &arr[i]);
  }
                                                                                               }
                                                                                            }
}
void insertionSort(int arr[], int n)
{
                                                pid_t pid = fork();
  int i, key, j;
                                                if (pid < 0) {
  for (i = 1; i < n; i++) {
                                                   fprintf(stderr, "Fork
    key = arr[i];
    j = i - 1;
                                              failed\n");
                                                   exit(EXIT_FAILURE);
     // Move elements of arr[0..i-
                                                } else if (pid == 0) {
1] that are greater than key to
                                                   printf("Child process sorting
one position ahead of their
                                              using Insertion Sort.\n");
current position
                                                   insertionSort(arr, n);
     while (j \ge 0 \&\& arr[j] > key)
                                                   printf("Sorted array in child
{
                                              process:\n");
       arr[j + 1] = arr[j];
                                                   for (int i = 0; i < n; i++) {
                                                      printf("%d", arr[i]);
       j = j - 1;
     }
                                                   }
```