## WEEK 4

Write a C program to simulate Real-Time CPU Scheduling algorithms:

- a) Rate- Monotonic
- b) Earliest-deadline First

```
CODE:
#include <stdio.h>
#include <stdlib.h>
int et[10], i, n, dl[10], p[10], ready[10], flag = 1;
int lcm(int a, int b) {
  int max = (a > b)? a : b;
  while (1) {
     if (\max \% a == 0 \&\& \max \% b == 0)
        return max;
     max++;
  }
}
int lcmArray(int arr[], int n) {
  int result = arr[0];
  for (int i = 1; i < n; i++) {
     result = lcm(result, arr[i]);
  }
  return result;
}
void mono() {
  int time = lcmArray(dl, n);
  int op = 0, pr = 0, pre = pr;
  while (op <= time) {
     for (i = 0; i < n; i++) {
```

```
if (op % dI[i] == 0) {
     ready[i] = 1;
  }
}
flag = 0;
for (i = 0; i < n; i++) {
  if (ready[i] == 1) {
     flag = 1;
     break;
  }
}
if (flag == 0) {
  pr = -1;
} else {
  pr = -1;
  for (i = 0; i < n; i++) {
     if (ready[i] == 1) {
        if (pr == -1 || dl[i] < dl[pr]) {
           pr = i;
        }
     }
  }
}
  if (pr != pre) {
     if (pr == -1) {
        printf("%d Idle ",op);
     } else {
        printf("%d P%d ",op, pr + 1);
     }
  }
op++;
if (pr != -1) {
  p[pr] = p[pr] - 1;
  if (p[pr] == 0) {
     p[pr] = et[pr];
```

```
ready[pr] = 0;
        }
     }
     pre = pr;
  }
  printf("\n");
void edf() {
  int time = lcmArray(dl, n);
  int op = 0, pr = 0, pre = -1;
  int flag, i;
  while (op <= time) {
     for (i = 0; i < n; i++) {
        if (op % dI[i] == 0) {
           ready[i] = 1;
        }
     }
     flag = 0;
     for (i = 0; i < n; i++) {
        if (ready[i] == 1) {
           flag = 1;
           break;
        }
     }
     if (flag == 0) {
        pr = -1;
     } else {
        pr = -1;
        for (i = 0; i < n; i++) {
           if (ready[i] == 1) {
              if (pr == -1 || p[i] < p[pr]) {
                 pr = i;
```

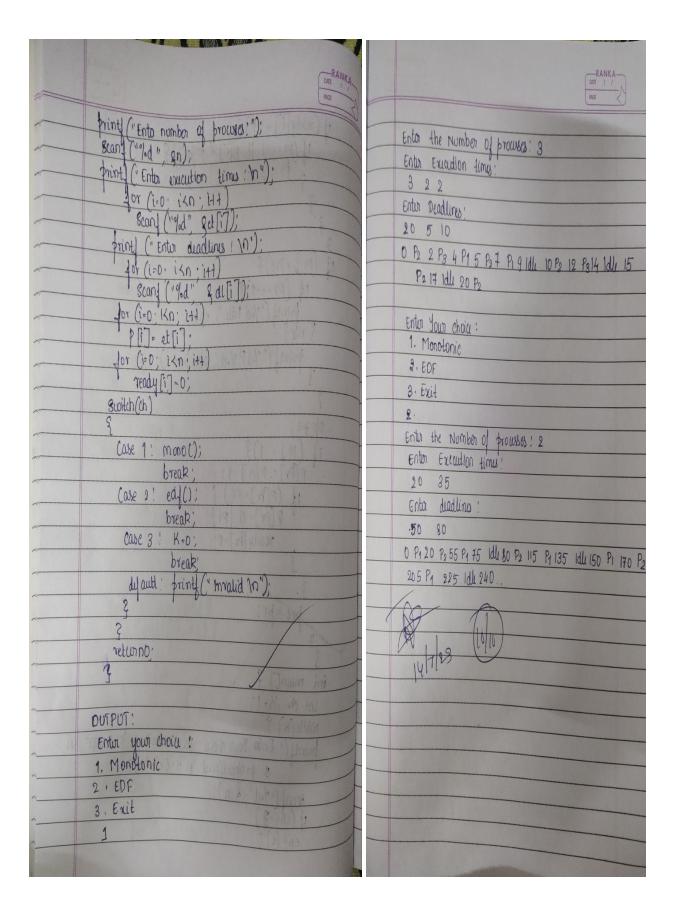
```
}
          }
        }
     }
     if (pr != pre) {
        if (pr == -1) {
           printf("%d Idle ", op);
        } else {
           printf("%d P%d ", op, pr + 1);
        }
     }
     op++;
     if (pr != -1) {
        p[pr] = p[pr] - 1;
        if (p[pr] == 0) {
           p[pr] = et[pr];
          ready[pr] = 0;
        }
     }
     pre = pr;
  printf("\n");
}
int main() {
  int ch, k = 1;
  while (k) {
     printf("Enter your choice: \n1. Monotonic \n2. EDF \n3. Proportional \n4. Exit\n");
     scanf("%d", &ch);
     if(ch==3)
     exit(0);
     printf("Enter the number of processes: ");
     scanf("%d", &n);
     printf("Enter execution times: \n");
     for (i = 0; i < n; i++)
```

```
scanf("%d", &et[i]);
     printf("Enter deadlines: \n");
     for (i = 0; i < n; i++)
        scanf("%d", &dl[i]);
     for (i = 0; i < n; i++)
        p[i] = et[i];
     for (i = 0; i < n; i++)
        ready[i] = 0;
     switch (ch) {
        case 1:
           mono();
           break;
        case 2:
           edf();
           break;
        case 3:
           k = 0;
           break;
        default:
           printf("Invalid choice.\n");
     }
  }
  return 0;
}
```

## **OBSERVATION:**

مالالما	ON ANKA ONE 3	OUT / PROFE
18/7/23		(111)
	Write a c program to simulate Real-Fine CPU	1 (op % dl [e] == 0) {
	Libert a a pringram to simulate that a corp	(c) (at [1] = 0) }
	Schiduling algorithms	ready [i] -1;
	a) Rati - Morotonic	
	a) Rati - Marotonic b) Earliest - dualline First	Hen O'
		109:0;
	#indudi <8tdio.h>	or (i=0; i <n: i++)="" td="" {<=""></n:>
	# Indud (Stalib.h)	1 (ready [7] ==1) {
	Thomas I was fall lime of	Jag = 1;
	int et [10], i, n, d [10], p[10], ready [10], flag = 1;	break;
		3
	int lcm (inta, intb)?	1 Al A A A
	int my = (u/o)	1 (100 == 6) }
	while (1)?	3 We 3 8 min San Mar
	tohile (1) }	3 We } { 1 mi >qu   lutat
	alian MAY!	pro-1; - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1 - 1
	mad ++; and many and all are	or (i=0; ixn, itt)}
		1] (ready[i] ==1) {
	3 00 000 10 170 15 000 1 178 0	14 (pr==-1    al[i] < al[pr]) {
	int lom Array (int arr), intn).	prel'
	the result = arr [0];	7 5
	for (ht i=1. Kn; H+) ?	3 4 (40 0) 0140
	result = 1 cm (result, arr[i]);	3
	7800 = 1011 (1800 1 - 101)	1 (py 1 = pxe) {
	3	1 (Pr==-1) {
	retian result;	print ( "/d dk " op);
	3	S elles
	void mono() }	print( 1/d P/d ) Op, prt1);
	int time = 1 cm Array (dl,n)	3
	int op=0, tr=0, tre-br	3 Paul P
	int time = 1 cm Array (dl,n);  Int op = 0, pr=0, pre=pr:  while (op<=time) {	optt;
	Jor (1-0; i <n; i+1)="" td="" }<=""><td>fill of about</td></n;>	fill of about
	1	

RANKA DOT NOE	RANKA—
1 (P) 1 =-1) {	1 (ready[i] ==1) {
Provi Providi	11 (Pres-1 11 pli7 < play 1) S
14 (P[Po] == b) {	1 (Pres-1    P[i] < P[P]) { Pres-1    P[i] < P[P]) {
P[Pr] - P[Pr] -1;    (P[Pr] = et[Pr];    P[Pr] = et[Pr];	25
1004 [A] = 0;	3
3	3 Common and Unit
3 2 14 1131	1) (Pr = Pre) S
pre apr;	14 (Pr==-1) {
	print ("%d Idle", op);
3030	} ule {
void eaf()}	print (" o/d P o/od", op, pr+1);
int time = lam Array (at, n),	3 Co-Malor
roid edf()?  int time = lom Array (dl, n);  int op=0, pr=0, pre=-1;  int flag, e;  while (op==time)?	(0) the
int (1093 8;	0)2+1,
while (op 2= time / 3	1 (PS = -1) } (100m 100) P[Pr] - P[Pr] -1; 4881
LONG (00) idn; ith) {	P[Pr] = P[Pr] -1;
13 (0) % all 1 = 20/2	1 (P[Pi] == 0) { (1603 (4 3/0)) P[Pi] = et   Pi]; (1600)
18014 1 1 2 1,	P(PT)= et (PT), 40001
2	transfer in the second
1100 01	3
	pre = pr;
11 (ready [1] ==1) {	nic Pi
11 (100-1)	2
lag=1; break;	int main() {
2	int ch ,K-1;
2	while (K) }
11 (1lag == 0) {	print! (" Ento Your choic: In 1. Monotonic In 9. EDF In
br = -1;	3. proportional in 4. Exit in);
3 Use {	srant ("% d" &ch);
pr=-1;	y (cherg)
or (i.o., i.h.) }	eut (o);



## **OUTPUT**:

```
■ "C:\Users\ysrmo\OneDrive - Base PU College\Desktop\4thsem\OS\oslab\edfrm\bin\Debug\edfrm.exe"
                                                                                                                                                                                 Ð
Enter your choice:
1. Monotonic
2. EDF
3. Exit
Enter the number of processes: 3
Enter execution times:
3 2 2
Enter deadlines:
20 5 10
0 P2 2 P3 4 P1 5 P2 7 P1 9 Idle 10 P2 12 P3 14 Idle 15 P2 17 Idle 20 P2
Enter your choice:
1. Monotonic
2. EDF
3. Exit
Enter the number of processes: 2
Enter execution times:
20 35
Enter deadlines:
50 80
0 P1 20 P2 55 P1 75 Idle 80 P2 115 P1 135 Idle 150 P1 170 P2 205 P1 225 Idle 240 P2 250 P1 270 P2 295 Idle 300 P1 320 P2 355 P1 375 Idle 400 P1
Enter your choice:
1. Monotonic
2. EDF
3. Exit
```