```c
/*###################################################################*/
/*                                                                   */
/*  HW06_PART1                                                       */
/* Tarih : 6.4.15                                                    */
/* Hazirlayan : HASAN MEN                                            */
/*                                                                   */
/*Enerji degerlerine gore iscilere yeni is atayan, gun ve haftanin  */
/* en iyi iscisini bulup dosyaya basan program parcacigi            */
/*                                                                   */
/*###################################################################*/

#include <stdio.h>

#define NUM_EMPLOYEES 4 /* Array satir sayisi - isci sayisi*/
#define NUM_DAYS 7  /* Array sutun sayisi - gun sayisi */

/* yeni veri tiplerimiz */
typedef enum{Ali,Ayse,Fatma,Mehmet} employee;   /* iscilerimiz */
typedef enum{Monday,Tuesday,Wednesday,Thursday,
                Friday,Saturday,Sunday} day_of_week;    /* haftanin gunleri */


/*###################################################################*/
/*  Dosyadan verileri okuyarak dizimize atama yapar                  */
/*Girdiler:                                                          */
/*  const char*file name = okunacak dosyanin stringdeki adi          */
/*Cikti:                                                             */
/*  m- output parametre olarak kullanilacak dizi                     */
/*###################################################################*/
void read_matrix(const char* file_name, int m[NUM_EMPLOYEES][NUM_DAYS]);

/*Arada olusan dizileri kontrol etmek icin ekrana basar */
void print_input(int m[NUM_EMPLOYEES][NUM_DAYS]);

/*###################################################################*/
/* Iscilerin onceki gunlerde yaptiklari islere bakarak cok is yapana*/
/* az enerjili isi verir.Ayni seviyede olan varsa ilkinden itibaren */
/* isleri atar.                                                      */
/* Girdi:                                                            */
/* -m[7][4] = read_matrix fonksiyonu ile doldurulan dizimiz          */
/* Cikti:                                                            */
/* -job_schedule[7][4] = iscilerin duzenlenmis yeni is haritasi      */
/*                                                                   */
/*NOT= Arrayler constant degiller uzerlerinde islemler yapilmistir  */
/*###################################################################*/
void create_work_plan(  int job_schedule[NUM_EMPLOYEES][NUM_DAYS],
                        int m[NUM_EMPLOYEES][NUM_DAYS]);

/* Bir dizi alir ve buble sort ile buyukten kucuge siralar */
void sort(int a[]);

/* Olusturulan tek boyutlu dizileri kontrol icin ekrana basar*/
void a1(int b[]);


/* Iscilerin onceki islerinin sirasini bulmamiza yarar. */
/* yapilan islem ornegi ekrana adim adim basilmistir    */
void ques(int a[],int b[]);

/* haftanin gunlerini kullanarak en iyi isciyi bulur    */
/* ve employee turunden return eder */
employee find_the_employer_of_the_day(int work_schedule[NUM_EMPLOYEES]
[NUM_DAYS], day_of_week day_name);

/* tum hafta yapilan islere gore en iyi isciyi return eder */
employee find_the_employer_of_the_week(int work_schedule[NUM_EMPLOYEES]
[NUM_DAYS]);

/* Tum islemler sonucu yeni arrayde olusan bilgileri dosyaya yazar */
void report(const char* file_name, int job_scheduling[NUM_EMPLOYEES][NUM_DAYS]);

/* employee find_the_employer_of_the_week'ten gelen isciyide dosyaya yazar */
```

```c
void print_name(employee best,FILE *oPtr);


int main()
{
    /* Ana fonksiyon baslangici */
    const char inP[]="Energies.txt";    /* input dosyamiz */
    const char outP[]="Report.txt";     /* output dosyamiz */
    int job_energies[NUM_EMPLOYEES][NUM_DAYS];  /* okuma yapilacak array */
    int schedule[NUM_EMPLOYEES][NUM_DAYS]={0};  /* siralanmis array */
    /* degiskenlerin sonu */

    read_matrix(inP,job_energies);  /* input dosyasindan arraya okuma yapilir*/
    print_input(job_energies);  /* dolu arrayin ekrana basilmasi */

    /* rastgele alinan islerin duzenlenmesi */
    create_work_plan(schedule,job_energies);
    print_input(schedule);  /* duzenlenmis islerin ekrana basilmasi */

    report(outP,schedule);  /* tum bilgilerin dosyaya yazilmasi */

    return 0;
}

void read_matrix(const char* file_name, int m[NUM_EMPLOYEES][NUM_DAYS])
{
    int status,i,j; /* local degiskenlerimiz */

    FILE *file=fopen(file_name,"r"); /* dosyanin acilmasi */

    if(file==NULL)  /* dosya acilmaz ise ata mesaji ver */
    {
        printf("########\nFile couldn't opened to read!!!\n");
        printf("Results failed!!!\n########\n");
    }
    else
    {
        /* dosya sonuna kadar okunan degerleri array oturt */
        do{

            for(i=0;i<NUM_DAYS;i++)
            {
                for(j=0;j<NUM_EMPLOYEES;j++)
                {
                    status=fscanf(file,"%d",&m[j][i]);
                }
            }

        }while(status!=EOF);

    /* dosyanin kapanmasi */
    fclose(file);
    }
}

/* 4satir 7sutun olarak ekrana basilir */
void print_input(int m[NUM_EMPLOYEES][NUM_DAYS])
{

    int i,j; /* local degiskenler */
    printf("------------The content of Array---------------\n");
    for(i=0;i<NUM_EMPLOYEES;i++)
    {
        for(j=0;j<NUM_DAYS;j++)
            printf("%4d ",m[i][j]);
        printf("\n");
    }

    printf("\n---------------------------------------------\n");


}
```

```

void create_work_plan(  int job_schedule[NUM_EMPLOYEES][NUM_DAYS],
                        int m[NUM_EMPLOYEES][NUM_DAYS])
{

    int sums[4];    /* toplam enerji degeri */
    int days[4];    /* iscilerin gun icindeki islemleri  */
    int i;      /*sayac olarak kullanildi */
    int que[4]; /* sums taki degerlerin siralari */
    int day;    /* gunumuz */
    /* degiskenlerin sonu*/


    /* ILK GUN default olarak girildi  */
    for(i=0;i<4;i++)
        days[i]=m[i][0];

    sort(days);

    for(i=0;i<4;i++)
    {
        job_schedule[i][0]=days[i];
        sums[i]=days[i];
    }

    /* 1.gunden yani salidan itibaren bi onceki gunleri suma atiyorum   */
    /* sumdaki degerlerin sirasini buluyorum bu siralara gore yeni gelen */
    /* gunde ki degeleri (onceden sortlanmis) capraz esitliyorum */
    /**/
    for(day=1;day<NUM_DAYS;day++)       /* geri kalan 6 gun icin */
    {
        for(i=0;i<4;i++)
            days[i]=m[i][day];  /* 4isci icin enerjiler x=i icin girildi */

        sort(days); /* enerjiler buble sort edildi */

        ques(sums,que); /* sumlarin hangi sirada olduklarini*/
                        /* bulup ona gore dagitim yaptim */
        a1(que);    /* kontrol icin siralari ekrana bastim*/

        for(i=0;i<4;i++)
        {
            job_schedule[i][day]=days[que[i]-1];    /* yeni dizi dolduruldu */
            sums[i]+=job_schedule[i][day];/*sum degeri yeni gun ile toplandi*/
        }
    }
}

/* buble sort yapar */
void sort(int a[])
{
    int i,j,temp;

    for(i=0;i<NUM_EMPLOYEES;i++)
    {
        for(j=0;j<NUM_EMPLOYEES-1;j++)
        {
            if(a[j]<=a[j+1])
            {
                temp=a[j];
                a[j]=a[j+1];
                a[j+1]=temp;
            }
        }
    }
}

void a1(int b[])
{
    int i;

    for(i=0;i<NUM_EMPLOYEES;i++)
```

```c
        printf("Sum_que[%2d]=%2d ",i,b[i]);
    printf("\n");

}

/* Selection sorta benzer sekilde her eleman kac taneden buyuk esit */
/* diye bakar ve bunlari sayaca atar.Ayrica ayni elemanlar olmasina */
/* karsilik kendinden sonra ayni eleman geldiyse bunlarin sayisini  */
/* sayactan cikarip output dizisine atama yapar                     */

void ques(int a[],int b[])  /* SUMLARIN SIRASINI OGREN*/
{


    int i,j,sayac=0,esit=0; /* local degiskenler */
    for(i=0;i<4;i++)    /* tek tek tum elemanlar uzerinde gezeriz */
    {
        for(j=0;j<4;j++)
        {
            if(a[i]>=a[j])
            {
                sayac++;
            }
        }

        for(j=i+1;j<4;j++)
        {
        /* kendisinden sonra gelen ayni sayilari ignore ettik*/
            if(a[i]==a[j])
                esit++;
        }
        b[i]=sayac-esit;
        esit=0; /* yeni degeler icin degiskenler sifirlandi */
        sayac=0;
    }
}

employee find_the_employer_of_the_day(int work_schedule[NUM_EMPLOYEES]
[NUM_DAYS], day_of_week day_name)
{
    int i;
    int status=0;
    int max=0;
    /* local degiskenler */
    for(i=0;i<NUM_EMPLOYEES;i++)
    {
        if(work_schedule[i][day_name]>=max)
        {
            max=work_schedule[i][day_name]; /* en buyuk olani buluruz */
            status=i;   /* en buyuk olanini sirasini tutariz */
        }
    }

    return status;  /* gunun iscisi return edilir */
}

employee find_the_employer_of_the_week(int work_schedule[NUM_EMPLOYEES]
[NUM_DAYS])
{
    int i,j;
    int sum=0;  /* tum gunlerde yapilan toplam is */
    int max=0;
    int maxS=0; /* hafta ici en cok calisanin employee turunde numarasi */
    /* local degiskenler */


    for(i=0;i<NUM_EMPLOYEES;i++)
    {
        for(j=0;j<NUM_DAYS;j++)
        {
            sum+=work_schedule[i][j];
        }
        if(sum>max)
        {
            max=sum;
            maxS=i;
        }
        sum=0;
    }
        return maxS;
}

void report(const char* file_name, int job_scheduling[NUM_EMPLOYEES][NUM_DAYS])
{
    int i,j;
    employee best_day; /* gunun en iyisi */
    employee best_week; /* haftanin en iyisi */
    day_of_week day_t;  /* day degiskenimiz */
    FILE *out=fopen(file_name,"w");
    /* local degiskenler */

    fprintf(out,"%10cMonday Tuesday Wednesday"
    " Thursday Friday Saturday Sunday\n",' ');

    /* iscilerin toplam sure zarfinda islerinin dosyaya basilmasi */
    for(i=0;i<NUM_EMPLOYEES;i++)
    {
        print_name(i,out);
        for(j=0;j<NUM_DAYS;j++)
            fprintf(out,"%7d ",job_scheduling[i][j]);
        fprintf(out,"\n");
    }

    for(day_t=Monday;day_t<=Sunday;day_t++) /*hafta basindan sonuna en iyiler*/
    {

    best_day=find_the_employer_of_the_day(job_scheduling,day_t);

        switch(day_t)
        {
        case Monday: fprintf(out,"\nThe best employer of Monday:"); break;
        case Tuesday: fprintf(out,"\nThe best employer of Tuesday:"); break;
        case Wednesday: fprintf(out,"\nThe best employer of Wednesday:"); break;
        case Thursday: fprintf(out,"\nThe best employer of Thursday:"); break;
        case Friday: fprintf(out,"\nThe best employer of Friday:"); break;
        case Saturday: fprintf(out,"\nThe best employer of Saturday:"); break;
        case Sunday: fprintf(out,"\nThe best employer of Sunday:"); break;
        }
        print_name(best_day,out);   /* gunlerden sonra isci isimleri basilir*/

    }

    /* haftanin en iyisinin yazilmasi */
    best_week=find_the_employer_of_the_week(job_scheduling);
    fprintf(out,"\nThe best employer of the week is ");
    print_name(best_week,out);
    fprintf(out,"Congratulation ");
    print_name(best_week,out);

    /* dosyanin kapanmasi */
    fclose(out);
}
/* isimleri tek tek yazmak yerine switch ile her yerde zorlanmadan yazariz */
void print_name(employee best,FILE *oPtr)
{
    switch(best)
        {
            case Ali: fprintf(oPtr,"%s%5c","Ali",' '); break;
            case Ayse: fprintf(oPtr,"%s%4c","Ayse",' '); break;
            case Fatma: fprintf(oPtr,"%s%3c","Fatma",' '); break;
            case Mehmet: fprintf(oPtr,"%s%2c","Mehmet",' '); break;
        }
}
/* HW06_HASAN_MEN_131044009_part1.c SONU */
```