```c
#ifndef _sirvid_h
#define _sirvid_h

#include <stdio.h>
#include <time.h>
#include <math.h>

typedef struct {
    int paev;
    int kuu;
    int aasta;
} kuupaev;

void prindiKuupaev(char [], kuupaev);
kuupaev saaHetkeneKuupaev(void);
kuupaev gregoorius_sirviks(kuupaev);
int onSirviPysivPyha(kuupaev);
int saaPooripaevad(int [3], int);
int onLiigaasta(int);
int intArvAastas(int);
int intArvKuus(int, int);
int intGregooriusVahe(kuupaev, kuupaev);
int intGregooriusOnSuurem(kuupaev, kuupaev);
int intAastaArvVahe(int, int);
int intAastasTulemasArv(kuupaev);
int intAastasOlnudArv(kuupaev);
double dblArvTaiskuuni(kuupaev algK);
kuupaev saaKuupaev_lisaArv(kuupaev, double);
double dblYmardaArv(double);
int onVordneGregoorius(kuupaev, kuupaev);
void saaSirviLiikuvadPyhad(kuupaev [10], int, int *);
double dblArvKuutsyklini(kuupaev, kuupaev, double);
double dblArvViimaseVeerandini(kuupaev);
double dblArvKuuLoomiseni(kuupaev);
double dblArvEsimeseVeerandini(kuupaev);
kuupaev saaKuupaev_lisaArv2(kuupaev, double);
int onPekopaev(kuupaev k);
int onSirviLiikuvPyha(kuupaev k, int *);
int onKuupaev(kuupaev);
kuupaev saaTaiskuu_kuupaev(kuupaev, int *);
kuupaev saaKuuLoomise_kuupaev(kuupaev, int *);
kuupaev saaKuuEsimeseVeerandi_kuupaev(kuupaev, int *);
kuupaev saaKuuViimaseVeerandi_kuupaev(kuupaev, int *);
int onVordneKuupaev(kuupaev, kuupaev);
int intSaaNadalapaevKuupaevast(kuupaev);

/*int onVordneGregoorius(kuupaev k1, kuupaev k2)
Tagastab kas kuupaevad on identsed või mitte.
*/
int onVordneKuupaev(kuupaev k1, kuupaev k2)
{
    return(k1.paev == k2.paev && k1.kuu == k2.kuu && k1.aasta == k2.aasta);
}
int intSaaNadalapaevKuupaevast(kuupaev k)
{
    kuupaev esmaspaev;
    esmaspaev.paev = 4;
```

```
    esmaspaev.kuu = 1;
    esmaspaev.aasta = 2010;

    int p = intGregooriusVahe(esmaspaev, k);

    if(p > 0)
        p = p % 7;
    else if(p < 0)
    {
        p *= -1;
        p = p % 7;
        if(p > 0)
            p = 7 - p;
    }
    p++;
    return(p);
}
/*dblYmardaArv(double arv)
Ümarda arv kuupäva lisa jaoks. Ümardus reeglid on järgnevad:
Kui koma koht on >= 0.6, siis ceilib numbri, muuljuhul floorib.
*/
double dblYmardaArv(double arv)
{
    double ret = floor(arv), abi = arv - ret;
    if(abi >= 0.6) return(ret + 1);
    return(ret);
}
/*kuupaev saaKuupaev_lisaArv(kuupaev k, double arv)
Lisab kuupäevale k number arv korda päevi (ümardatud dblYmardaArv(double arv)-uga).
Eeldatake, et kuupäev on Gregooriuse kuupäev. Kui arv on negatiivne, siis lahutatakse arv
korda päevi.
*/
kuupaev saaKuupaev_lisaArv(kuupaev k, double arv)
{
    double i;
    int intKuu = intArvKuus(k.kuu, k.aasta), onNeg = 0;
    if(arv < 0) onNeg = 1;
    if(onNeg) arv *= -1;
    arv = dblYmardaArv(arv) + 1;
    for(i = 1.0; i < arv; i++)
    {
        if(onNeg)
        {
            if(k.paev == 1)
            {
                if(k.kuu == 1)
                {
                    k.kuu = 12;
                    k.aasta--;
                }else
                    k.kuu--;
                k.paev = intArvKuus(k.kuu, k.aasta);
            }else
                k.paev--;
        } else {
            if(k.paev == intKuu)
            {
```

```
                k.paev = 1;
                if(k.kuu == 12)
                {
                    k.kuu = 1;
                    k.aasta++;
                }else
                    k.kuu++;

                intKuu = intArvKuus(k.kuu, k.aasta);
            }else
                k.paev++;
        }

    }
    return(k);
}

kuupaev saaKuupaev_lisaArv2(kuupaev k, double arv)
{
    double i;
    int intKuu = intArvKuus(k.kuu, k.aasta), onNeg = 0;
    if(arv < 0) onNeg = 1;
    if(onNeg) arv *= -1;
    arv = floor(arv) + 1;
    for(i = 1.0; i < arv; i++)
    {
        if(onNeg)
        {
            if(k.paev == 1)
            {
                if(k.kuu == 1)
                {
                    k.kuu = 12;
                    k.aasta--;
                }else
                    k.kuu--;
                k.paev = intArvKuus(k.kuu, k.aasta);
            }else
                k.paev--;
        } else {
            if(k.paev == intKuu)
            {
                k.paev = 1;
                if(k.kuu == 12)
                {
                    k.kuu = 1;
                    k.aasta++;
                }else
                    k.kuu++;

                intKuu = intArvKuus(k.kuu, k.aasta);
            }else
                k.paev++;
        }

    }
    return(k);
```

```
}
double dblArvKuutsyklini(kuupaev baasK, kuupaev k, double dblKorrigeerija)
{
    int intSuhe = intGregooriusOnSuurem(baasK, k);
    double ret = 0.0, dblTsykkel = 29.53;
    if(intSuhe == 0) return(ret);
    ret = (double) ((intGregooriusVahe(baasK, k) * intSuhe) + dblKorrigeerija) / dblTsykkel;
    ret -= floor(ret);
    if(intSuhe == 1)
        ret = 1.0 - ret;
    ret *= dblTsykkel;
    return(ret);
}
double dblArvTaiskuuni(kuupaev k)
{
    kuupaev taiskuuK;
    taiskuuK.paev = 30;
    taiskuuK.kuu = 1;
    taiskuuK.aasta = 2010;
    return(dblArvKuutsyklini(taiskuuK, k, 0.39));
}
double dblArvKuuLoomiseni(kuupaev k)
{
    kuupaev loomineK;
    loomineK.paev = 15;
    loomineK.kuu = 1;
    loomineK.aasta = 2010;

    return(dblArvKuutsyklini(loomineK, k, 0.43));
}
double dblArvEsimeseVeerandini(kuupaev k)
{
    kuupaev esimeneVeerandK;
    esimeneVeerandK.paev = 23;
    esimeneVeerandK.kuu = 1;
    esimeneVeerandK.aasta = 2010;
    return(dblArvKuutsyklini(esimeneVeerandK, k, 0.57));
}
double dblArvViimaseVeerandini(kuupaev k)
{
    kuupaev viimaneVeerandK;
    viimaneVeerandK.paev = 7;
    viimaneVeerandK.kuu = 1;
    viimaneVeerandK.aasta = 2010;
    return(dblArvKuutsyklini(viimaneVeerandK, k, 0.57));
}
/*int intAastaArvVahe(int a1, int a2)
Arvutab aastate vahe paevades (kaasaarvatud päevad aastates a1 ning a2).
NB! - Eeldatakse, et a1 <= a2
*/
int intAastaArvVahe(int a1, int a2)
{
    int i, ret = 0;
    a2 += 1;
    for(i = a1; i < a2; i++)
        ret += intArvAastas(i);
    return(ret);
```

```
}
/*int intAastasTulemasArv(kuupaev k)
Tagastab päevade arvu aastas, mis on veel Tulemas. (hetkest kuupäeva arvestatakse, kui juba
olnud päev ehk ei võeta arvesse)
*/
int intAastasTulemasArv(kuupaev k)
{
    return(intArvAastas(k.aasta) - intAastasOlnudArv(k));
}
/*int intAastasOlnudArv(kuupaev k)
Tagastab aastas olnud päevade arvu, kaasaarvatud hetkese päeva.
*/
int intAastasOlnudArv(kuupaev k)
{
    int ret = 0, i;
    for(i = 1; i < k.kuu; i++)
        ret += intArvKuus(i, k.aasta);
    return(ret + k.paev);
}
/*int intGregooriusOnSuurem(kuupaev k1, kuupaev k2)
Võrdleb kaht kuupäeva k1 ja k2 ning väljastab, kumb on suurem või väiksem või kas need on
võrdsed.
Tagastab:
0 - Kui kuupäevad k1 ja k2 on võrdsed ehk k1 == k2
1 - Kui kuupäevad k2 on suurem kui k1 ehk k2 > k1
-1 - Kui kuupäevad k1 on suurem kui k2 ehk k2 < k1
*/
int intGregooriusOnSuurem(kuupaev k1, kuupaev k2)
{

    if(k1.aasta != k2.aasta)
    {
        if(k1.aasta < k2.aasta)
            return(1);
        else
            return(-1);
    }else if(k1.kuu != k2.kuu) {
        if(k1.kuu < k2.kuu)
            return(1);
        else
            return(-1);
    }else if(k1.paev != k2.paev)
        {
            if(k1.paev < k2.paev)
                return(1);
            else
                return(-1);
        }
    return(0);
}
/*int intGregooriusVahe(kuupaev k1, kuupaev k2)
Tagastab vahe päevades, kahe kuupäeva vahel. Eeldatakse, et kuupäevad on Gregooriuse
kalendri kuupäevad.

Sammuti võib öelda, mitme päevaga saab k1-st k2-te. seega kui k1 = 02.01.1999 ja k2 =
04.01.1999, siis vahe nende kuupäevade vahel on 2, kuna liidad päeva k1-le, saad 03.01.1999,
kuid lisad 2 päeva k1-le, siis saad 04.01.1999
```

```
*/
int intGregooriusVahe(kuupaev k1, kuupaev k2)
{

    int ret = 0, intVahe, suhe = intGregooriusOnSuurem(k1, k2);
    kuupaev abiK;
    if(suhe == 0) return(ret);
    if(suhe == -1) { abiK = k2; k2 = k1; k1 = abiK; }

    intVahe = k2.aasta - k1.aasta;
    if(intVahe > 1)
        ret = intAastaArvVahe(k1.aasta + 1, k2.aasta - 1) + intAastasTulemasArv(k1) +
intAastasOlnudArv(k2);
    else if(intVahe == 1)
        ret = intAastasTulemasArv(k1) + intAastasOlnudArv(k2);
    else
        ret = intArvAastas(k1.aasta) - intAastasTulemasArv(k2) - intAastasOlnudArv(k1);

    if(suhe == -1) ret *= -1;
    return(ret);
}
int intArvAastas(int intAasta)
{
    return(365 + onLiigaasta(intAasta));
}
int intArvKuus(int intKuu, int intAasta)
{
    switch(intKuu)
    {
        case 4:
        case 6:
        case 9:
        case 11:
            return(30);
        case 2:
            return(28 + onLiigaasta(intAasta));
    }
    return(31);
}
int onLiigaasta(int intAasta)
{
    if(intAasta % 4 == 0 && intAasta % 100 != 0 || intAasta % 400 == 0) return(1);
    return(0);
}
int saaPooripaevad(int intRetArr[3], int intAasta)
{
    FILE *a;
    int aasta;
    a = fopen("txt/pooripaevad.txt", "r");

    if(a != NULL)
    {
        while(!feof(a))
        {
            fscanf(a, "%d", &aasta);
            fscanf(a, "%d", &intRetArr[0]);
            fscanf(a, "%d", &intRetArr[1]);
```

```
            fscanf(a, "%d", &intRetArr[2]);
            if(aasta == intAasta)
            {
                fclose(a);
                return(1);
            }
        }
        fclose(a);
    }

    return(0);
}
void saaSirviLiikuvadPyhad(kuupaev kArr[10], int aasta, int *onTeada)
{
    int pArr[3], paev;
    *onTeada = saaPooripaevad(pArr, aasta);
    kuupaev
        kevadPooripaev,
        suvePooripaev,
        sygisPooripaev,
        kihlakud,
        eidepaev,
        urbepaev,
        munapyha,
        maahengaus,
        suvisted;

    kevadPooripaev.aasta = aasta;
    kevadPooripaev.paev = 20;
    kevadPooripaev.kuu = 3;

    suvePooripaev.aasta = aasta;
    suvePooripaev.paev = 21;
    suvePooripaev.kuu = 6;

    sygisPooripaev.aasta = aasta;
    sygisPooripaev.paev = 22;
    sygisPooripaev.kuu = 9;
    if(*onTeada)
    {
        kevadPooripaev.paev = pArr[0];
        suvePooripaev.paev = pArr[1];
        sygisPooripaev.paev = pArr[2];
    }

    munapyha = saaKuupaev_lisaArv(kevadPooripaev, dblArvTaiskuuni(kevadPooripaev));
    paev = intSaaNadalapaevKuupaevast(munapyha);
    if(paev != 7)
        munapyha = saaKuupaev_lisaArv(munapyha, 7 - paev);

    if(munapyha.paev == 25 && munapyha.kuu == 3)
        munapyha = saaKuupaev_lisaArv(munapyha, 28.0);

    maahengaus = saaKuupaev_lisaArv(munapyha, 39.0);
    suvisted = saaKuupaev_lisaArv(munapyha, 49.0);
    urbepaev = saaKuupaev_lisaArv(munapyha, -7.0);
    kihlakud = saaKuupaev_lisaArv(munapyha, -47.0);
```

```
    eidepaev = saaKuupaev_lisaArv(kihlakud, 22.0);

    kArr[0] = kevadPooripaev;
    kArr[1] = suvePooripaev;
    kArr[2] = sygisPooripaev;
    kArr[3] = munapyha;
    kArr[4] = maahengaus;
    kArr[5] = suvisted;
    kArr[6] = urbepaev;
    kArr[7] = kihlakud;
    kArr[8] = eidepaev;
    kArr[9] = saaKuupaev_lisaArv(kihlakud, 1.0);
}
int onSirviLiikuvPyha(kuupaev k, int *onTeada)
{
    kuupaev kArr[10];
    int i;
    saaSirviLiikuvadPyhad(kArr, k.aasta, onTeada);
    for(i = 0; i < 11; i++)
        if(onVordneKuupaev(k, kArr[i]))
        {
            if(i != 9) return(i + 1);
            else return(8);
        }
    return(0);
}
int onSirviPysivPyha(kuupaev k)
{
    switch(k.kuu)
    {
        case 1:
            if(k.paev == 14) return(1);
            if(k.paev == 17) return(2);
            break;
        case 2:
            if(k.paev == 2) return(3);
            if(k.paev == 9) return(4);
            break;
        case 3:
            if(k.paev == 9) return(5);
            if(k.paev == 25) return(6);
            break;
        case 4:
            if(k.paev == 14) return(7);
            if(k.paev == 23) return(8);
            break;
        case 5:
            if(k.paev == 9) return(9);
            break;
        case 6:
            if(k.paev == 23) return(10);
            break;
        case 7:
            if(k.paev == 2) return(11);
            if(k.paev == 13) return(12);
            if(k.paev == 29) return(13);
            break;
```

```
        case 8:
            if(k.paev == 10) return(14);
            if(k.paev == 15) return(15);
            if(k.paev == 24) return(16);
            break;
        case 9:
            if(k.paev == 8) return(17);
            if(k.paev == 29) return(18);
            break;
        case 10:
            if(k.paev == 14) return(19);
            break;
        case 11:
            if(k.paev == 2) return(20);
            if(k.paev == 10) return(21);
            if(k.paev == 25) return(22);
            break;
        case 12:
            if(k.paev == 7) return(23);
            if(k.paev == 25) return(24);
            if(k.paev > 20 && k.paev < 25) return(25);
            break;
    }
    if(onPekopaev(k))
        return(26);
    return(0);
}
int onPekopaev(kuupaev k)
{
    if(k.kuu == 9)
    {
        kuupaev k2 = k;
        k2.paev = 1;
        k2 = saaKuupaev_lisaArv(k2, (double) (7 - intSaaNadalapaevKuupaevast(k2)));
        k2 = saaKuupaev_lisaArv(k2, 14.0);
        if(onVordneKuupaev(k2, k))
            return(1);
    }
    return(0);
}
kuupaev gregoorius_sirviks(kuupaev greKuupaev)
{
    greKuupaev.aasta = greKuupaev.aasta + 8213;
    return greKuupaev;
}
kuupaev saaHetkeneKuupaev(void)
{
    time_t timer = time(NULL);
    struct tm *aeg = localtime(&timer);
    kuupaev nyyd;
    nyyd.paev = aeg->tm_mday;
    nyyd.kuu = aeg->tm_mon + 1;
    nyyd.aasta = aeg->tm_year + 1900;
    return nyyd;
}
void prindiKuupaev(char selgitus[], kuupaev a)
{
```

```
    if(a.kuu > 9 && a.paev > 9)
    {
        printf("%s %d.%d.%d\n", selgitus, a.paev, a.kuu, a.aasta);
    }

    else if(a.kuu > 9 && a.paev < 10)
    {
        printf("%s 0%d.%d.%d\n", selgitus, a.paev, a.kuu, a.aasta);
    }
    else if(a.kuu < 10 && a.paev > 9)
    {
        printf("%s %d.0%d.%d\n", selgitus, a.paev, a.kuu, a.aasta);
    }
    else
    {
        printf("%s 0%d.0%d.%d\n", selgitus, a.paev, a.kuu, a.aasta);
    }

}
int onKuupaev(kuupaev k)
{
    if(k.kuu > 0 && k.kuu < 13 && k.paev > 0)
        if(k.paev <= intArvKuus(k.kuu, k.aasta)) return(1);
    return(0);
}
kuupaev saaTaiskuu_kuupaev(kuupaev k, int *onTeada)
{
    kuupaev k1, k2;
    FILE *f;
    int intVahe, intVahe2;
    *onTeada = 0;

    k1 = saaKuupaev_lisaArv2(k, dblArvTaiskuuni(k));

    f = fopen("txt/taiskuu.txt", "r");
    if(f != NULL)
    {
        while(!feof(f))
        {
            fscanf(f, "%d", &k2.aasta);
            fscanf(f, "%d", &k2.kuu);
            fscanf(f, "%d", &k2.paev);

            intVahe = intGregooriusVahe(k1, k2);
            intVahe2 = intGregooriusVahe(k, k2);

            if(intVahe > -5 && intVahe < 5 || intVahe2 > -5 && intVahe2 < 5)
            {
                *onTeada = 1;
                fclose(f);
                return(k2);
            }
        }
        fclose(f);
    }
    return(k1);
```

```
}
kuupaev saaKuuLoomise_kuupaev(kuupaev k, int *onTeada)
{
    kuupaev k1, k2;
    FILE *f;
    int intVahe, intVahe2;
    *onTeada = 0;

    k1 = saaKuupaev_lisaArv2(k, dblArvKuuLoomiseni(k));

    f = fopen("txt/kuu_loomine.txt", "r");
    if(f != NULL)
    {
        while(!feof(f))
        {
            fscanf(f, "%d", &k2.aasta);
            fscanf(f, "%d", &k2.kuu);
            fscanf(f, "%d", &k2.paev);

            intVahe = intGregooriusVahe(k1, k2);
            intVahe2 = intGregooriusVahe(k, k2);

            if(intVahe > -5 && intVahe < 5 || intVahe2 > -5 && intVahe2 < 5)
            {
                *onTeada = 1;
                fclose(f);
                return(k2);
            }
        }
        fclose(f);
    }
    return(k1);
}
kuupaev saaKuuEsimeseVeerandi_kuupaev(kuupaev k, int *onTeada)
{
    kuupaev k1, k2;
    FILE *f;
    int intVahe, intVahe2;
    *onTeada = 0;

    k1 = saaKuupaev_lisaArv2(k, dblArvEsimeseVeerandini(k));

    f = fopen("txt/esimene_veerand.txt", "r");
    if(f != NULL)
    {
        while(!feof(f))
        {
            fscanf(f, "%d", &k2.aasta);
            fscanf(f, "%d", &k2.kuu);
            fscanf(f, "%d", &k2.paev);

            intVahe = intGregooriusVahe(k1, k2);
            intVahe2 = intGregooriusVahe(k, k2);

            if(intVahe > -5 && intVahe < 5 || intVahe2 > -5 && intVahe2 < 5)
            {
                *onTeada = 1;
```

```
                fclose(f);
                return(k2);
            }
        }
        fclose(f);
    }
    return(k1);
}
kuupaev saaKuuViimaseVeerandi_kuupaev(kuupaev k, int *onTeada)
{
    kuupaev k1, k2;
    FILE *f;
    int intVahe, intVahe2;
    *onTeada = 0;

    k1 = saaKuupaev_lisaArv2(k, dblArvViimaseVeerandini(k));

    f = fopen("txt/viimane_veerand.txt", "r");
    if(f != NULL)
    {
        while(!feof(f))
        {
            fscanf(f, "%d", &k2.aasta);
            fscanf(f, "%d", &k2.kuu);
            fscanf(f, "%d", &k2.paev);

            intVahe = intGregooriusVahe(k1, k2);
            intVahe2 = intGregooriusVahe(k, k2);

            if(intVahe > -5 && intVahe < 5 || intVahe2 > -5 && intVahe2 < 5)
            {
                *onTeada = 1;
                fclose(f);
                return(k2);
            }
        }
        fclose(f);
    }
    return(k1);
}
#endif
```