

# DIM0108.0

## Correção dos exercícios da aula 26

David Deharbe [DIMAp/UFRN]

22 de novembro de 2011

**Aviso.** Devido à limitação de tempo disponível, as soluções apresentadas neste documento não foram compiladas nem testadas e provavelmente incluem erros.

*Defina um tipo para representar datas (do calendário).*

Uma data é definida com três números: o ano, o mês e o dia. Esses números são inteiros positivos. Seleccionamos a construção de registro para definir o tipo. Estes registros terão sempre três campos:

```
typedef struct TDate {
    unsigned d; /* dia 1..31 */
    unsigned m; /* mes 1..12 */
    unsigned a; /* ano */
} Tdate;
```

*Defina uma rotina que imprime uma data, usando o formato DD/MM/AAAA, na saída padrão.*

A definição da rotina é direta. Tomamos cuidado em observar o formato de dois dígitos para o dia e o mês, e de quatro dígitos para o ano.

```
#include <stdio.h>
void date_print_ddmmaaaa(Tdate date)
{
    printf("%02u/%02u/%04u", date.d, date.m, date.a);
}
```

*Defina uma rotina que imprime uma data em português, na saída padrão. O formato usado deve ser o mesmo do cabeçalho desta folha.*

Para desenvolver esta rotina, devemos ter uma forma de determinar o nome em português de cada mês do ano. Para isto, lançaremos mão de um arranjo de 12 textos, um para cada mês. Já que estes nomes são pré-determinados, podemos atribuir o valor desejado já na declaração do arranjo, através da inicialização do mesmo:

```
#include <stdio.h>

char * month_pt [12] = {
    "janeiro", "fevereiro", "marco",
    "abril", "maio", "junho",
    "julho", "agosto", "setembro",
    "outubro", "novembro", "dezembro"};
```

Uma vez definido o arranjo `month_pt`, a definição da rotina solicitada é simples. Deve-se tomar o cuidado de decrementar o número do mês para acessar o arranjo, já que as posições válidas vão de 0 até 11, e os meses vão de 1 até 12.

```
void date_print_pt (Tdate date)
{
    printf("%u de %s de %u", date.d, month_pt[date.m - 1], date.a);
}
```

*Escreva um programa, com uma variável do tipo usado para representar data, esta variável deve ser atribuída a data do seu nascimento, e então o seu valor deve ser impresso em português na saída padrão.*

Segue um programa que seria escrito por alguém nascido em 21 de maio de 1968.

```
#include <stdio.h>

typedef struct Tdate {
    unsigned d;
    unsigned m;
    unsigned a;
} Tdate;

int main (void)
{
    Tdate birthdate;
    birthdate.d = 21;
    birthdate.m = 5;
    birthdate.a = 1968;
    print_date_pt (birthdate);
    return 0;
}
```

*Defina uma rotina que lê uma data, no formato DD/MM/AAAA, da entrada padrão, e retorna esta data.*

```
Tdate date_read_ddmmaaaa (void)
{
    Tdate date;
    scanf("%u/%u/%u", &date.d, &date.m, &date.a);
    return date;
}
```

*Defina uma rotina que tem como parâmetro duas datas, retorna -1 se a primeira for anterior à segunda, 0 se são iguais, e 1 se a primeira for posterior à segunda.*

```
int date_compare (Tdate date1, Tdate date2)
{
    if (date1.a == date2.a && date1.m == date2.m && date1.d == date2.d)
        return 0;
    if (date1.a < date2.a ||
        (date1.a == date2.a && date1.m < date2.m) ||
        (date1.a == date2.a && date1.m == date2.m && date1.d < date2.d))
        return -1;
    return 1;
}
```

*Defina uma rotina que tem como parâmetro uma data, e retorna a data do dia seguinte.*

Esta rotina tem que levar em conta três situações:

- O dia é o último do ano; neste caso o dia seguinte é o 1º de janeiro do ano seguinte. Essa situação é determinada com o valor dos campos representando o dia e o mês, que devem ser 31 e 12 respectivamente.
- O dia é o último dia do mês (sem ser de dezembro); neste caso o dia seguinte é o dia 1º do mês seguinte do mesmo ano. Essa situação é um pouco complicada a determinar, haja vista que o número do último dia do mês varia para cada mês do ano, e ainda deve levar em conta o caso do mês de fevereiro dos anos bissextos, que possui um número de dias diferente. Para determinar esta situação, definimos a rotina `leap_year` que determina se um ano é bissexto, e uma rotina `days_in_month` que retorna o número de dias que há no mês. Esta rotina usa um arranjo com o número de dia de cada mês e ajusta o resultado quando o ano é bissexto e o mês é fevereiro.
- Para todos os outros dias, o dia seguinte é obtido incrementando o número do dia atual, permanecendo iguais o mês e o ano.

```
int leap_year (int a)
{
    return a % 400 == 0 || (a % 4 == 0 && a % 100 != 0);
}

int days_in_month_tab[12] =
    { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

int days_in_month(int m, int a)
{
    int result;
    assert(1 <= m && m <= 12);
    result = days_in_month_tab[m-1];
    if (m == 2 && leap_year(a))
        result += 1;
    return result;
}

Tdate date_next (Tdate date)
{
    Tdate result = date;
    if (date.m == 12 && date.d == 31) {
        result.a = date.a+1;
        result.m = 1;
        result.d = 1;
    } else if (date.d == days_in_month(date.m, date.y)) {
        result.a = date.a;
        result.m = date.m + 1;
        result.d = 1;
    } else {
        result.a = date.a;
        result.m = date.m;
        result.d = date.d + 1;
    }
}
```

```

return result;
}

```

*Defina uma rotina que tem como parâmetro uma data, e indica qual a posição daquela data no ano vigente (começando com a posição 1 para o dia 1º de janeiro).*

Para calcular qual a posição de um dia no ano, basta somar a posição do dia no mês atual com a soma do número de dias dos meses anteriores, o qual pode ser obtido chamando a rotina `days_in_month`.

```

int date_position(Tdate date)
{
    int m;
    int result;
    result = date.d;
    for (m = 1; m <= date.m - 1; ++m)
        result += days_in_month(date.m, date.a);
    return result;
}

```

*Defina uma rotina que tem como parâmetro duas datas, sendo a primeira nunca posterior à segunda, e indica quantos dias ocorreram entre as duas datas. Se as datas são iguais, a rotina deve retornar 0, se a segunda data foi o dia seguinte à primeira, a rotina deve retornar 1, e assim em diante.*

Para calcular o número de dias entre duas datas  $d_1$  e  $d_2$ , assumindo que  $d_1$  é anterior a  $d_2$ , determinamos três quantidades de dias:

- $t_1$  é a posição de  $d_1$  no ano de  $d_1$ .
- $t_2$  é a posição de  $d_2$  na ano de  $d_2$ .
- $t_3$  é a quantidade de dias entre o 1º de janeiro do ano de  $d_1$  e o número de dias entre o 1º de janeiro de  $d_2$ .

O resultado é simplesmente calculado com a expressão  $t_2 + t_3 - t_1$ . Para calcular  $t_1$  e  $t_2$ , podemos usar a rotina `date_position`. Para calcular  $t_3$ , somamos o número de dias em cada ano de  $d_1$  até o ano anterior a  $d_2$ . O número de dias é 366 nos anos bissextos, e 365 nos outros anos. Definimos uma rotina `days_in_year` que, dada um ano, retorna esta informação.

```

/*
Assume date1 = d1/m1/a1 and date2 = d2/m2/a2.
Consider the following durations
                                <----result----->
01/01/a1<--t1-->d1/m1/a1<----->01/01/a2<--t2-->d2/m2/a2
                                <-----t3----->
we have:
result = t2+t3-t1 and
t1 = date_position(date1) and
t2 = date_position(date2) and
t3 = SUM{a : a1..a2-1} days_in_year(a)
*/
int days_in_year (int a)
{
    if (leap_year(a))
        return 366;
    else

```

```

        return 365;
    }

int date_diff(Tdate datel, date2)
{
    int a;
    int result;
    assert(date_compare(datel, date2) != 1);
    result = 0;
    for (a = datel.a; a < date2.a; ++a)
        result += days_in_year(a);
    result += date_position(date2);
    result -= date_position(datel);
    return result;
}

```

*Escreva um programa que calcula quantos dias uma pessoa já viveu. O programa deve ler a data de nascimento da pessoa, a data de hoje, e imprimir o resultado.*

Este programa pode ser desenvolvido combinando de forma direta os tipos e rotinas definidos anteriormente:

```

#include <stdio.h>

int main(void)
{
    Tdate today;
    Tdate birthdate;
    int days;
    printf("Enter your birth date (DD/MM/YYYY):");
    birthdate = date_read_ddmmaaaa();
    printf("Enter today's date (DD/MM/YYYY):");
    today = date_read_ddmmaaaa();
    days = date_diff(birthdate, today);
    printf("You have lived %i days.\n", days);
    return 0;
}

```