Computação para Informática - 2016/2
Primeira Prova - 24 de outubro de 2016

---

1. (25 Pontos) Escreva um programa que apresente a sequencia mostrada no exemplo abaixo.

   Entrada

   Não há nenhuma entrada neste problema.

   Saída

   Imprima a sequência conforme exemplo abaixo.

Entrada	Saída
	I=1 J=7 I=1 J=6 I=1 J=5 I=3 J=9 I=3 J=8 I=3 J=7 ... I=9 J=15 I=9 J=14 I=9 J=13

```c
/*
 * 1097.c
 *
 * Copyright 2016 Adriano Cruz <adriano@nce.ufrj.br>
 *
 */
#include<stdio.h>

int main (void) {
     int i, j;

     for (i = 1; i <= 9; i=i+2) {
          for (j = i+6; j > i+6-3; j--) {
               printf("I=%d J=%d\n", i , j);
          }
     }
}
```

2. (25 Pontos) Números perfeitos são números que são iguais a soma dos seus divisores próprios, excluindo o próprio número. Vamos considerar que um número ($dp > 0$) é divisor próprio de um número N quando o resto da divisão de N por dp é zero. Segundo a definição vamos considerar que N não é divisor próprio de si mesmo. Por exemplo, o primeiro número perfeito é o 6, já que a soma de seus divisores próprios é igual a 6, ou seja $6 = 1 + 2 + 3$. O segundo número perfeito é $28 = 1 + 2 + 4 + 7 + 14$.

   Tarefa

   Escreva um programa que descubra todos os números perfeitos até 10000.

   Entrada

   Não há entrada

   Saída

   Imprima os números perfeitos um por linha. Ao final imprima o total de números encontrados.

Entrada	Saída
	1 6 28 496 8128 5

```c
/*
 * perfeitos.c
 *
 * Copyright 2016 Adriano Cruz <adriano@nce.ufrj.br>
 *
 */

# include <stdio.h>

int main() {
     int i, j, soma;
     int total = 0;

     for (i = 1; i <= 10000; i++) {
          soma = 1;
          for (j = 2; j <= i/2; j++ )  {
               if (i % j == 0) soma += j;
          }
          if (soma == i) {
               printf("%d \n", i);
               total++;
          }
     }

     printf("%d\n", total);
     return 0;
}
```

3. (25 Pontos) Zé Sá quer montar um painel de leds contendo diversos números. Ele não possui muitos leds, e não tem certeza se conseguirá montar o número desejado. A Figura 1 mostra quantos leds são gastos para montar cada um dos algarimos. Cada retângulo vermelho corresponde a um led. Por exemplo, para montar o algarismo 3 são necessários 5 leds. Considerando esta configuração, escreva um programa que ajude João a descobrir a quantidade de leds necessário para montar o valor desejado.

Entrada

A entrada contém um inteiro $N, (1 \leq N \leq 1000)$ correspondente ao número de casos de teste, seguido de N linhas, cada linha contendo um número $(1 \leq V \leq 10^{100})$ correspondente ao valor que João quer montar com os leds.

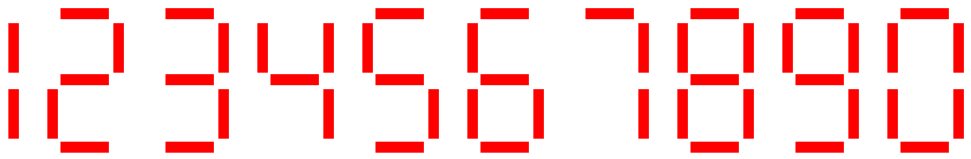

Figura 1: Configuração dos leds.

Saída

Para cada caso de teste, imprima uma linha contendo o número de leds que João precisa para montar o valor desejado, seguido da palavra "leds".

Entrada	Saída
3 115380 2819311 23456	27 leds 29 leds 25 leds

```c
/*
 * 1168.c
 *
 * Copyright 2016 Adriano Cruz <adriano@nce.ufrj.br>
 * Descricao:   Este programa calcula quantos leds sao necessarios
 *              para escrever um numero.
 *              Este problema tem varia solucoes. Esta aqui nao
 *              usa comandos de teste tais como: if ou switch.
 *              Ele cria um vetor inicializado com quantos leds
 *              sao necessarios para cada numero. Eu uso o numero
 *              para obter diretamente do vetor a quantidade de leds.
 */


#include <stdio.h>
#include <string.h>

int main(int argc, char **argv)
{
    int quantNumeros;
    int i, j;
    char numero[102];
    int leds[10] = {6, 2, 5, 5, 4, 5, 6, 3, 7, 6};
    int quantLeds;
    char temp[81];

    fgets(temp, 81, stdin);
    sscanf(temp, "%d", &quantNumeros);

    for(i = 0; i < quantNumeros; i++) {
        fgets(numero, 102, stdin);
        numero[strlen(numero)-1] = '\0';
        quantLeds = 0;
        for (j = 0; j < strlen(numero); j++) {
            quantLeds += leds[numero[j] - '0'];
        }
        printf("%d leds\n", quantLeds);
    }

    return 0;
}
```

4. (25 Pontos) Durante anos, todos os contratos da Associação de Servidores de Pindorama (ASP) foram datilografados (digitados) em uma velha máquina de datilografia. Recentemente Sr. Miranda, um dos contadores da ASP, percebeu que a máquina apresentava falha em um, e apenas um, dos algarismos do teclado. Mais especificamente, o dígito falho, quando datilografado, não é impresso na folha, como se a tecla correspondente não tivesse sido pressionada. Ele percebeu que isso poderia ter alterado os valores numéricos representados nos contratos. Preocupado com a contabilidade, ele quer descobrir, a partir dos valores originais negociados nos contratos, que ele mantém em anotações manuscritas, quais os valores de fato representados nos contratos. Por exemplo, se a máquina apresenta falha no dígito 5, o valor 1500 seria datilografado no contrato como 100, pois o 5 não seria impresso. Note que o Sr. Miranda quer saber o valor numérico errado representado no contrato, ou seja, nessa mesma máquina, o número 5000 corresponde ao valor numérico 0, e não 000 (como ele de fato aparece impresso).

Entrada

A entrada consiste de diversos casos de teste, cada um em uma linha. Cada linha contém dois inteiros D e N ($1 \leq D \leq 9, 1 \leq N < 10^{100}$), representando, respectivamente, o dígito que está apresentando problema na máquina e o número que foi negociado originalmente no contrato (que podem ser grande, pois Pindorama tem sido acometido por hiperinflação nas últimas décadas várias vezes).

O ultimo caso de teste é seguido por uma linha que contém apenas dois zeros separados por espaços em branco.

Saída

Para cada caso de teste da entrada o seu programa deve imprimir uma linha contendo um único inteiro V, o valor numérico representado de fato no contrato.

Entrada	Saída
5 5000000 3 123456 9 23454324543423 9 99999999991999999 7 777 5 5000001 0 0	0 12456 23454324543423 1 0 1

```c
/*
 * 1120.c
 *
 * Copyright 2016 Adriano Cruz <adriano@nce.ufrj.br>
 *
 */


#include <stdio.h>
#include <string.h>
#include <stdlib.h>

int main(int argc, char **argv)
{
    int defeito;
    char erro;
    char valor[101];
    char convertido[101];
    int i, j;
    int conta;

    while (1) {
        scanf("%d %s", &defeito, valor);
        erro = '0' + defeito;
        if ((erro == '0') && (valor[0] == '0')) break;
        for (i = 0, j = 0; i < strlen(valor); i++) {
            if (valor[i] != erro) {
                convertido[j] = valor[i];
                j++;
            }
        }
        convertido[j] = '\0';
        for (conta = 0, i = 0; i < strlen(convertido);  i++) {
            if (convertido[i] == '0') conta++;
        }
        if((conta == strlen(convertido))&&(convertido[0] == '0')) {
            convertido[0] = '\0';
        }
        if (strlen(convertido) == 0) {
            /* puts("0"); */
            printf("%lld\n", 0ll);
        }
        else {
            /* puts(convertido);
            printf("%lld\n", atoll(convertido)); */
            i = 0; while (convertido[i] == '0') i++;
            puts(&convertido[i]);
        }
    }

    return 0;
}
```