Universidade Federal de Goiás
Instituto de Informática

INF0286 – Algoritmos e Estruturas de Dados 1

Lista 1 – Revisão de Introdução à Programação

Prof. Dr. Aldo André Díaz Salazar

1 (⋆) Cometa

Figura 1: Cometa Halley

O cometa Halley é um dos cometas do Sistema Solar de menor período, completando uma volta em torno do Sol a cada 76 anos. Em 1986, na última ocasião em que ele se tornou visível desde o planeta Terra, várias agências espaciais enviaram sondas para coletar amostras de sua cauda e assim confirmar teorias sobre sua composição química.

Escreva um programa `C` que, dado o ano atual, determine qual é o próximo ano em que o cometa Halley será visível novamente. Se o ano atual é um ano de passagem do cometa, considere que o cometa já passou nesse ano, ou seja, considere sempre o próximo ano de passagem após o atual.

Considerações

- O ano de 1986 é o marco de sincronismo.
- Considere os anos bissextos, ou seja, a cada quatro anos há uma correção de um dia (no futuro e no passado) em relação ao ano solar de 365 dias.

Entrada

O ano base de cálculo, $0 \leq a \leq 3000$.

Saída

O próximo ano em que o cometa Halley será visível novamente.

Tabela 1: Exemplo – Anos de avistamento do Cometa Halley.

Entrada	Saída
2635	2670
2010	2062
2270	2290
1910	1986
460	465

2 ($\star$) José

José, que começou a ir à escola, está tendo problemas com os números. Para ajudá-lo a "*pegar o jeito*", a professora de José escreve dois números de três dígitos e pede ele para comparar esses números. Entretanto, ao invés de interpretá-los com o dígito mais significativo à esquerda, ele deve interpretá-los com o dígito mais significativo à direita. Ele tem que dizer à professora qual o maior dos dois números.

Escreva um programa em `C` que seja capaz de verificar as respostas de José.

Entrada

Dois números diferentes de três dígitos.

Saída

O maior dos números da entrada, comparados conforme descrito no enunciado. O número deve ser escrito invertido, para mostrar a José como ele deve lê-lo e, havendo zeros à esquerda do número, eles não devem ser escritos, como apresentado nos seguintes exemplos.

Tabela 2: Exemplo – Comparação entre números.

Entrada	**Saída**
483 583	385
493 583	394
493 589	985
160 720	61
100 200	2

3 ($\star\star$) Stack overflow

Os computadores digitais foram inventados para realizar cálculos muito rapidamente e, atualmente, atendem a esse requisito de maneira extraordinária. Porém, nem toda "conta" pode ser feita num computador digital, pois ele não consegue representar todos os números possíveis dentro de um único endereço de memória.

Em um computador pessoal atual, por exemplo, o maior inteiro que é possível representar na unidade de memória *stack* é 18 446 744 070 000 000 000 ($2^{64} - 1$). Caso algum cálculo executado pelo computador dê um resultado acima desse número, ocorrerá um *overflow*, que é quando o resultado não pode ser representado por ser maior do que o valor máximo permitido.

Por exemplo, imagine um computador que somente pode representar números menores a 1024 ($2^{10} - 1$) e mandarmos ele executar a conta $1022 + 5$. Ocorrerá um *overflow*, já que o resultado deste cálculo será maior que 1023.

Elabore um programa `C` que seja capaz de receber o maior número que um computador consegue representar em memória e uma expressão de soma ou de multiplicação entre dois inteiros positivos. Determine se ocorrerá, ou não, *overflow* naquele computador.

Entrada

- Um inteiro representando o maior número que o computador consegue representar.
- Dois número inteiros a serem operados

- Um caractere, '+' ou '×', representando adição ou multiplicação.

Saída

A frase `overflow` se o resultado causar um *overflow*, ou `no overflow` caso contrário.

Tabela 3: Exemplo – *Stack overflow*.

Entrada	Saída
57 20 x 3	overflow
10 5 + 6	overflow
10 5 + 4	no overflow
57 12 x 3	no overflow
30 4 x 4	no overflow

4 (⋆⋆) Capicua

O pequeno estudante Alan Mathison Turing está aprendendo a decompor um número em unidades, dezenas, centenas, unidade de milhar, dezena de milhar, etc. Ele está com grandes dificuldade neste processo. Sua professora, Ada Lovelace, preocupada com o rendimento de Alan decidiu ensiná-lo por meio de uma brincadeira que utiliza os números *capicua* (também conhecidos por *palíndromos*):

Alan deve pegar um número com quatro dígitos e verificar se o reverso deste número é ele próprio. Se for, Alan deve responder `sim`, do contrário responde `nao`.

Escreva um programa em `C` que implemente esta brincadeira conforme descrita a seguir.

Entrada

- Um inteiro $n \geq 1$ representando a quantidade de números que Alan deve analisar.
- n números inteiros de até quatro dígitos.

Saída

As frases `sim` ou `nao` que correspondende à resposta.

Tabela 4: Exemplo – Números capicua.

Entrada	Saída
2 4569 5775	nao sim
3 1458	nao nao sim

1228 9779	
4 1221 2222 3113 7887	sim sim sim sim
5 1234 1243 1324 2134 4321	nao nao nao nao nao
10 1234 5897 1881 2662 5588 9229 3653 5555 3773 4587	nao nao sim sim nao sim nao sim sim nao

5 (⋆⋆) Computação

A capacidade do ser humano para calcular, foi um dos fatores que possibilitaram o desenvolvimento da Matemática, da Lógica e, por conseguinte, da Computação. Nos primórdios da Matemática e da Álgebra, utilizavam-se os dedos das mãos para efetuar cálculos, daí a origem da palavra *dígito*.

Por volta do século III a.C., o matemático indiano Pingala inventou o sistema de numeração binário, que hoje é utilizado no processamento dos computadores digitais. O sistema binário estabelece que sequências específicas de 1's (uns) e 0's (zeros) podem representar informações de números, letras, imagens, etc.

Figura 2: Pingala.

Recentemente, a BioSymbol (BS), startup criada por alunos do INF/UFG, inventou um computador de base 4 (tetrade), inspirado nas quatro bases nitrogenadas da Biologia (Adenina, Citosina, Guanina, Timina). A BS contratou você para fazer um programa em `C` que receba números inteiros positivos na base decimal e os converta para a base 4 utilizando divisões sucessivas.

Considerações

Utilize os símbolos A, C, G e T na base 4 para representar os dígitos 0, 1, 2 e 3, respectivamente.

Entrada

- A quantidade de números, $1 \leq n \leq 100$, que serão convertidos.
- n números inteiros não negativos.

Saída

Os valores correspondentes na base 4 para cada número digitado.

Tabela 5: Exemplo – Números capicua.

Entrada	Saída
5 1 2 3 4 10	C G T CA GG
2 16 8	CAA GA
9 1 2 3 4 10 16 8 5 11	C G T CA GG CAA GA CC GT
5 10 20 30 40 50	GG CCA CTG GGA TAG
10 1 2 3 4 1 2 3 4 1 2	C G T CA C G T CA C G

6 (⋆⋆) Envelopes

Kurt Gödel é um garoto muito esperto e que adora promoções e sorteios. Como já participou de muitas promoções da forma "para participar, envie n rótulos de produtos ...", Kurt tem o hábito de guardar o rótulo de todos os produtos que compra prevendo a possibilidade de ocorrência de promoções futuras. Dessa forma, sempre que uma empresa faz uma promoção, ele já tem "um monte" de rótulos para mandar. A Balas e Mababicodz Ltda (BM) está fazendo uma nova promoção, e, como era de se esperar, Kurt quer participar dela. É preciso enviar um envelope contendo um rótulo de cada tipo de bala que a BM produz.

Por exemplo, se a BM disser que produz três tipos de balas (A, B, C) e uma pessoa tem três rótulos de A, três rótulos de B e dois rótulos de C, ela pode enviar no máximo dois envelopes, já que falta um rótulo de C para compor o terceiro envelope. Sabe-se que não há limite para o número de envelopes que uma pessoa pode enviar.

Kurt gosta muito das balas e, por causa disso, a quantidade de rótulos que ele tem é muito grande. Porém, ele não está conseguindo determinar a quantidade máxima de envelopes que pode enviar. Como você conhece o Kurt, ele pediu sua ajuda para fazer o cálculo, de modo que ele compre o número exato de envelopes necessários para enviar para a promoção.

Escreva um programa em `C` que, a partir da lista de rótulos de Kurt, calcule o número máximo de envelopes válidos que ele pode enviar para a promoção.

Entrada

- A quantidade de rótulos de balas que Kurt possui, $1 \leq n \leq 1000$.
- O número de tipos diferentes de bala que a BM produz, $1 \leq k \leq 20$.
- Uma lista de n inteiros representando cada rótulo de bala que Kurt possui.

Saída

O número máximo de envelopes válidos que Kurt pode enviar.

Tabela 6: Exemplo – Envelopes.

Entrada	Saída
10 2 1 1 1 1 1 2 2 2 2 2	5
20 5 1 2 3 4 1 2 3 4 1 2 3 4 5 1 2 3 4 5 4 4	2
10 3 1 2 3 1 2 3 1 2 3 1	3
20 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	20

7 ($\star\star\star$) Sapos

Gustavo Silva, apelidado de GS, passou as suas férias no sítio do seus avós. Uma das suas atividades prediletas era nadar no rio que havia no fundo da casa. O que o mais impressionava do rio, era um belo caminho feito inteiramente com pedras. Há muito tempo, o avô de GS construiu este caminho com pedras posicionadas em linha reta. Ele tomou muito cuidado para que o espaçamento entre as pedras fosse de exatamente de um metro. Hoje em dia, a única utilidade do caminho é servir de diversão para os sapos que vivem no rio, que pulam de uma pedra a outra agitadamente.

Um certo dia, GS assistiu atentamente as acrobacias dos anfíbios e notou que cada sapo sempre pulava uma quantidade fixa de metros. GS sabe que você participa de hackathons e resolveu desafiá-lo com o seguinte problema:

"Cada pedra é identificada como 1, 2, 3, ..., a partir da sua posição a partir da margem do rio. Dado o número de pedras no rio, o número de sapos, a pedra sobre a qual cada sapo está e, a distância que cada sapo pula, determine as posições onde pode existir pelo menos um sapo depois que GS chega no rio após assistir o balé de pulos dos sapos".

Entrada

- O número de pedras no rio, $1 \leq p \geq 50$.
- O número de sapos, $1 \leq s \leq 100$.
- A posição inicial de cada sapo e sua distância de pulo.

Saída

- A possibilidade de cada pedra ter um sapo. Imprimir '1' para as pedras que podem ter um sapo. Imprimir '0' para as pedras que não podem ter um sapo.

Tabela 7: Exemplo – Sapos.

Entrada	Saída
5 2 3 2 4 4	1 0 1 1 1
8 3	0 1 1 1 0 1 0 1

3 3	
2 2	
6 2	
10 8	1 1 1 1 1 1 1 1 1 0
1 7	
2 5	
3 4	
4 7	
5 2	
6 9	
7 2	
8 3	
16 7	1 1 1 1 1 1 1 0 1 0 1 1 1 1 1 1
1 8	
2 7	
3 6	
4 5	
5 4	
6 3	
7 2	
10 10	1 1 1 1 1 1 1 1 1 1
1 1	
2 1	
3 1	
4 1	
5 1	
6 1	
7 1	
8 1	
9 1	
10 1	

8 (⋆⋆) Primos

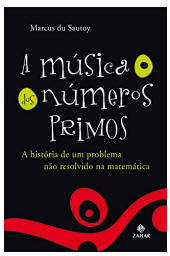

Figura 3: Livro de (SABOY, 2007).

No livro *A música dos números primos*, de Marcus du Saboy, o autor mostra que o mistério dos números primos passou a ser considerado o maior problema matemático de todos os tempos. Em meados do século XIX, o matemático alemão Georg Riemann formulou a seguinte hipótese:

"É possível estabelecer uma harmonia entre esses números primos, à semelhança da harmonia musical."

A partir de então, as mentes mais ambiciosas da Matemática embarcaram nessa procura que parece não ter fim. Atualmente, estipulou-se o prêmio de um milhão de dólares para quem provar a hipótese. O livro relata este santo graal da Matemática, com casos interessantes e retratos pintorescos dos personagens que, desde Euclides, se envolveram nesse estranho mistério.

Você deverá implementar em `C`, um algoritmo que seja capaz de identificar se um certo número é ou não um número primo. Números não primos são ditos de compostos.

Considerações

O seu programa deve estar preparado para receber números no intervalo de 2 a $2^{64} - 1$.

Entrada

- A quantidade de números de entrada, $1 \leq n \leq 100$.
- n números inteiros positivos a avaliar.

Saída

Responder *primo* ou *composto* para cada um dos números.

Tabela 8: Exemplo – Primos.

Entrada	**Saída**
5	
2	primo
3	primo
11	primo
16	composto
60	composto
9	
1200	composto
1697	primo
2712	composto
2549	primo
4723	primo
7853	primo
23557	primo
23558	composto
15485863	primo
10	
23	primo
29	primo
31	primo
37	primo
73	primo
79	primo
83	primo
101	primo
103	primo
107	primo
10	
24	composto
30	composto
32	composto
38	composto

74	composto
80	composto
84	composto
102	composto
104	composto
108	composto

9 ($\star\star\star$) Vetores

Uma operação comum em diversas áreas da computação científica é a multiplicação de números inteiros positivos com grande número de dígitos. Por exemplo, multiplicar um número de 24 dígitos por outro de 16 dígitos, o que pode gerar um número de até 40 dígitos.

Você está participando de uma equipe de desenvolvimento de uma aplicação científica que deve implementar, utilizando *vetores* para representar os números envolvidos, a operação de multiplicação mencionada. A aplicação deve ser desenvolvida utilizando a linguagem `C`, conforme a seguir especificado.

Entrada

- O número de casos de teste a serem aplicados, $1 \leq t \leq 50$.
- t linhas, cada uma contendo os dois números inteiros a serem multiplicados. Sabe-se que eles terão no máximo 40 dígitos cada e que poderão ser iguais a 0 (zero).

Saída

O resultado das operações de multiplicação dos pares de números correspondentes, na ordem em que foram fornecidos.

Tabela 9: Exemplo – Vetores.

Entrada	Saída
1 9423891297239 123857601272	1167220570724098896488008
2 9423891297239 123857601272 737238112845712940348123 1934720871365475	1167220570724098896488008 142634996408869612785857851064886325342 5
6 0 104759 0 104801 0 105331 104743 0 104789 0 104987 0	0 0 0 0 0 0
2 17546 92130935 737238112845712940348123 1934720871365475	1616529385510 142634996408869612785857851064886325342 5
2	

0 92130935	0
737238112845712940348123 0	0

10 ($\star\star\star$) Área de polígonos

Uma amiga sua está cursando Arquitetura e pediu auxílio para resolver o seguinte problema: Ela precisa calcular a área, em metros quadrados, de diversos polígonos:

- Círculo ('c'), cujo raio é dado por r
- Elipse ('e'), cujos raios maior e menor são r_M e r_m
- Triângulo ('t'), cujos lados são a, b e c
- Trapézio ('z'), cujas bases maior e menor são b_M e b_m, e a altura é h

Você pensou elaborar um programa em `C` que seja capaz de receber as informações necessárias e retorne a área do polígono.

Considerações

- Utilize $\pi = 3.14159265$.
- Os parâmetros dos polígonos são números inteiros positivos.

Entrada

- A quantidade de polígonos, $n \geq 1$.
- n linhas descrevendo o caractere que identifica cada polígono e seus respectivos parâmetros.

Saída

A área de cada polígono, arredondada, e sem casas decimais de precisão.

Tabela 10: Exemplo – Área de polígonos.

Entrada	Saída
4 c 2 e 2 4 t 8 8 8 z 7 3 4	 13 25 28 20
4 c 5 e 3 7 t 3 4 5 z 7 10 4	 79 66 6 34
3 t 3 4 5 t 5 5 5	 6 11

t 6 8 10	24
3 e 5 5 e 4 8 e 1 2	79 101 6
3 t 2 2 2 t 4 4 4 t 9 9 9	2 7 35

11 ($\star\star$) Matrizes 1

Um recurso matemático extremamente utilizado em Computação é o de *matriz*. Uma matriz de ordem $M \times N$ tem M linhas e N colunas. Escreva um programa em `C` que atenda às especificações indicadas a seguir:

Entrada

- A ordem da matriz, com $M \geq 1$ e $N \leq 10$.
- Uma operação matricial, entre '+' (adição) ou '$\times$' (multiplicação).
- Os elementos das matrizes, com valores entre -50 e $+50$.

Saída

Uma matriz contendo o resultado da operação. Lembre-se que há regras para a multiplicação matricial. Caso não seja possível realizá-la, o programa deverá implementar e sinalizar a transposição da segunda matriz para poder operá-la.

Tabela 11: Exemplo – Matrizes 1.

Entrada	**Saída**
2 3 1 2 3 4 5 6 + 6 5 4 3 2 1	7 7 7 7 7 7
2 3 1 2 3 4 5 6 x 6 5 4 3 2 1	28 10 73 28 TRANSPOSICAO
2 2 1 2 4 5	12 9 39 30

```
   x
  6 5
  3 2
-------------------------
   4 4
1 0 0 0     1 0 0 1
0 1 0 0     0 1 1 0
0 0 1 0     0 1 1 0
0 0 0 1     1 0 0 1
   +
0 0 0 1
0 0 1 0
0 1 0 0
1 0 0 0
-------------------------
   4 4
1 0 0 0     0 0 0 1
0 1 0 0     0 0 1 0
0 0 1 0     0 1 0 0
0 0 0 1     1 0 0 0
   x
0 0 0 1
0 0 1 0
0 1 0 0
1 0 0 0
-------------------------
```

12 ($\star\star\star\star$) Matrizes 2

Considere apenas matrizes bidimensionais. Escreva um programa em `C` que atenda às especificações indicadas a seguir:

Entrada

- A ordem da matriz, com $M \geq 1$ e $N \leq 10$.
- Uma operação matricial, entre 'I' (inversão de matriz), 'T' (transposição de matriz), ou 'D' (determinante da matriz).
- Os elementos da matriz, com valores entre -50 e $+50$.

Saída

Uma matriz contendo o resultado da operação. Lembre-se que há regras para a inversão de matriz e o cálculo do determinante. Se não for possível realizar alguma operação, sinalizar o erro na saída.

Tabela 12: Exemplo – Matrizes 2.

Entrada	**Saída**
2 3	1 4

```
1 2 3
4 5 6
  T
```

```
2 5
3 6
```

```
  2 3
1 2 3
4 5 6
  D
```

```
ERROR
```

```
  2 2
  1 2
  4 5
  D
```

```
-3
```

```
  4 4
2 0 0 0
0 2 0 0
0 0 2 0
0 0 0 2
  I
```

```
0.5 0 0 0
0 0.5 0 0
0 0 0.5 0
0 0 0 0.5
```

```
  4 4
0 0 0 1
0 0 1 0
0 1 0 0
1 0 0 0
  D
```

```
1
```

---

13 (⋆⋆) Números de Fibonacci

O matemático italiano Leonardo Fibonacci (1170–1250) foi de grande influência na Idade Média, sendo por muitos considerado como o maior deste período. Aos 32 anos, publicou o livro "Liber Abaci" (o Livro do Ábaco, ou Livro de Cálculo), responsável pela disseminação dos números hindu-arábicos na Europa. Em Pisa, há uma estátua em sua homenagem, localizada na galeria ocidental do Camposanto, pelos grandes serviços que ele prestou à cidade.

Figura 4: Estátua de Fibonacci.

Fibonacci também descobriu uma curiosa sequência numérica, posteriormente batizada como a *sequência de Fibonacci* e os números que a formam de *números de Fibonacci*, os quais são definidos da seguinte maneira:

$$
\begin{aligned}
s_0 &= 0 \\
s_1 &= 1 \\
s_2 &= s_0 + s_1 = 1 \\
s_3 &= s_1 + s_2 = 2 \\
s_4 &= s_2 + s_3 = 3 \\
s_5 &= s_3 + s_4 = 5 \\
s_6 &= s_4 + s_5 = 8 \\
&\vdots \\
s_n &= s_{n-1} + s_{n-2}
\end{aligned}
$$

Escreva, em `C`, um programa que receba o valor de n, com $3 \leq n \leq 100$, e escreva o valor de s_n. Seu programa deverá prever números de Fibonacci extremamente grandes, por exemplo, $s_{100} = 354224848179261915075$.

Entrada

- O número de casos de teste, $1 \leq k \leq 10$.
- Os valores de n.

Saída

Os valores calculados para o n correspondente.

Tabela 13: Exemplo – Números de Fibonacci.

Entrada	**Saída**
4	
3	2
4	3
5	5
6	8
7	
7	13
8	21
9	34
10	55
11	89
12	144
13	233

14 ($\star\star\star\star$) Fatoração de números de Fibonacci

Continuando com o estudo do exercício anterior, o que se deseja agora é apresentar a fatoração de um certo número de Fibonacci. Você deverá escrever, em `C`, um programa que receba o valor de n e imprima a fatoração dos s_n solicitados.

Entrada

- O número de casos de teste, $1 \leq k \leq 10$.
- Um valor para n, $1 \leq k \leq 100$.

Saída

Os fatores de cada s_n, apresentados em ordem estritamente crescente.

Tabela 14: Exemplo – Fatoração de números de Fibonacci.

Entrada	**Saída**
4	
3	2
4	3
5	5
6	8 = 2 x 2 x 2
7	
7	13
8	21 = 3 x 7
9	34 = 2 x 17
10	55 = 5 x 11
11	89
12	144 = 2 x 2 x 2 x 2 x 3 x 3
13	233
1	
100	354224848179261915075 = 3 x 5 x 5 x 11 x 41 x 101 x 151 x 401 x 3001 x 570601

Note que no último exemplo tem-se $s_{100} = 354224848179261915075$, cuja fatoração é $3 \times 5 \times 5 \times 11 \times 41 \times 101 \times 151 \times 401 \times 3001 \times 570601$.