Exercícios para aprender algoritmos

Por Pedro Kantek
Universidade Federal do Paraná
Curitiba - Brasil

e-mail: pkantek@gmail.com
www.pkantek.com.br

Introdução

Uma galeria de modelos e de exemplos de exercícios que podem ser usados em cursos de programação. Cada modelo é apresentado numa cor diferente, e esta cor dá nome à categoria dos exercícios.

Convencional

Descrição - O modelo preferidos dos livros de programação. Dá-se uma definição tanto detalhada quanto possível e o aluno deve escrever um programa que resolva o algoritmo pedido.

Resposta esperada - Um código a ser compilado e que deve gerar o resultado esperado.

Correção - Muito difícil e demorada. Usualmente o código não compila e a correção não pode ser na base binária: certo ou errado, pois isto geraria a quase totalidade de exercícios errados. O professor tem enorme dificuldade na correção: exceto nos casos trivais de programas muito simples, o resultado apresentado pelo aluno é quase impossível de interpretar. Uma alternativa seria e é orientar o aluno para que o computador faça a correção, mas neste caso, mais habilidades e mais desenvoltura ainda é exigida do aluno. Aceitável depois de meses ou anos, esta estratégia é quase impossível no início dos cursos de programação.

Este é o único modelo que não vai ser tratado aqui, uma vez que não exige uma programação prévia. É o usado desde sempre, não há muito que tratar. Ao contrário, neste documento se trata de exercícios e abordagens apoiados por software criado para ajudar ao professor e ao aluno.

Modelos Ad hoc

Na série de modelos a seguir, sempre há uma programação especialmente preparada para apoiar os exercícios propostos. Na verdade, este artigo descreve esta programação e como ela pode melhorar o desempenho dos alunos nos cursos. Em outras palavras, todos os modelos a seguir se apoiam em uma plataforma de software (feito e mantido pelo professor ou pela instituição) que apoia e ajuda nos exercícios e sobretudo em sua correção.

verde - Entender o algoritmo

Descrição - Descreve-se o problema e a sua solução. Apresentam-se exemplos numéricos resolvidos – o que ajuda a entender o funcionamento do algoritmo, usando-se a estratégia de *aprender pelo exemplo*. Apresentam-se uma ou poucas instâncias inéditas do problema. O aluno precisa seguir o algoritmo normalmente via lápis, embora os mais valentes podem pôr o computador a trabalhar e achar a resposta numérica esperada.

Resposta esperada - Um ou poucos números.

Correção - Simples e rápida, pode ser delegada a assistentes.

Infraestrutura Toda a programação VIVO já criada.

A seguir, um exemplo escolhido para apresentar esta classe. Não foi fácil a escolha há mais de 450 exercícios que poderiam ser escolhidos. É a maior classe. Veja o exemplo VIVO640 que expõe o algoritmo PERT-CPM e pede para o aluno calcular alguns parâmetros em um problema adhoc.

Método PERT-CPM

Em 1957 estávamos em plena Guerra Fria. Começava o projeto dos mísseis Polaris (250 empreiteiros, 9000 sub-empreiteiros, 70000 peças diferentes a fabricar - nunca antes fabricadas, atrasos no projeto inaceitáveis). Criou-se o método PERT (*Program Evaluation and Review Technique* - Técnica de avaliação e controle de programas). Ao mesmo tempo na Du Pont (indústria química) surgiu o método CPM (*Critical Path Method* - Método do Caminho Crítico). A partir de 1962, uma junção de ambos começou a ser conhecido como PERT-CPM.

Dado um projeto (conjunto de tarefas, suas interdependências e prazos, tendo como finalidade um determinado objetivo), o analista escreve uma rede. Nela, colocam-se as atividades e os eventos. Para cada atividade, determina-se a duração (exata ou probabilística) e a interdependência (eventos que a antecedem e que a sucedem).

Na rede, eventos conectam atividades e atividades conectam eventos. Existem diversas formas de representação, mas a mais comum é o método americano. Nele:

seta representa uma atividade. Por cima da seta escreve-se o nome da atividade e por baixo a sua duração.

circulo representa um evento. Dentro do círculo, o número ou nome do evento.

Quando duas atividades são feitas uma após a outra, apenas por comodidade, sem haver relação de dependência, os eventos em questão são interligados por uma atividade fantasma (uma seta tracejada). Atividades fantasma têm duração zero e podem, por conveniência ser deslocadas pela rede.

Dois eventos não podem ter mais de uma atividade entre eles. Se existirem 2 eventos realmente paralelos, introduz-se um evento fictício e uma atividade fantasma, a fim de obedecer à regra.

Quando uma atividade tem como pré-requisito uma condição externa à rede, (por exemplo, *tempo seco*), um evento fictício e uma atividade fantasma (tempo seco, neste caso) são introduzidas na rede.

Se a lista de tarefas é puramente linear e para começar uma a anterior deve ter terminado, não há muito que fazer e o desenho da rede tende a ser perda de tempo. Entretanto, se algum grau de paralelismo puder ser conseguido na realização de tarefas, então o método PERT-CPM tem muito a ajudar. No exemplo do missil Polaris, o uso da técnico baixou o tempo do projeto originalmente estimado para 5 anos, para pouco mais de 3.

Método

1. Liste as atividades do projeto. Se forem muitas, cogite trabalhar com macro-tarefas (e sub-redes associadas). Avalie a necessidade de quebrar tarefas ou, ao contrário, de juntá-las.

2. Estime a duração de cada tarefa. Escolha uma unidade adequada (dia, semana, mês ?) que seja única para toda a rede.

3. Desenhe a rede, começando no evento **início** e terminando no evento **fim**, da esquerda para a direita, dispondo tarefas (com nomes e duração) e eventos.

A questão agora, é **qual a duração total do projeto?** A resposta é a soma das tarefas que estiverem no pior caminho (em termos de duração) entre os eventos de início e fim. Este caminho (que pode não ser único) é chamado de **caminho crítico**. Quaisquer tarefas que estejam neste caminho e que sofram algum atraso certamente atrasarão todo o projeto. Em contrapartida, se elas sofrerem adiantamento, isto não garante que o projeto todo seja adiantado, pois agora, o caminho crítico poderá ser outro.

Para calcular o CC, atribui-se o tempo 0 (zero) ao evento início. Para cada atividade seguinte, soma-se o seu tempo com o tempo do(s) evento(s) precedente(s). Escreve-se o tempo de cada evento entre parênteses, sobre o círculo do evento. Este tempo também é chamado TEMPO MAIS CEDO ou simplesmente CEDO (early) e ele é o tempo necessário para que um evento seja atingido, desde que não haja atrasos nas atividades precedentes.

Se a um determinado evento chegam mais de uma atividade, comparam-se todas as somas possíveis e escolhe-se a maior delas.

Quando se obtiver o tempo do evento fim, este valor será a duração do projeto. O caminho que vai do início ao final, passando pelas atividades de maior tempo, é o caminho crítico.

As atividades não críticas (fora do caminho) podem eventualmente sofrer atraso sem que isto atrase o projeto.

Ao chegar ao último evento, a data cedo é a menor duração possível para o projeto. Esta data (apenas no último evento) deve ser copiada para baixo e passa a constituir o tempo mais tarde do último evento.

Outro tempo importante é o TEMPO MAIS TARDE, ou simplesmente TARDE (late) que é a data limite de realização de um evento. Qualquer execução que passar desta data, atrasará o projeto. O tempo TARDE é escrito em um pequeno retângulo sobre os parênteses do tempo CEDO.

Para calcular o tempo tarde, parte-se do final e para cada atividade que chega a ele, subtrai-se a duração das tarefas que a ele chegam. Se mais de uma tarefa sai de um evento, o tempo tarde é o menor valor dos resultados-subtrações.

Eventos que fazem parte do CC tem obrigatoriamente o CEDO = TARDE. Finalmente, a folga de um evento, é a diferença entre o TARDE - CEDO.

Exemplo

Seja a seguinte rede:

tarefa	origem	destino	duração
A	1	2	7
B	2	3	2
C	1	3	5
D	2	4	4
E	2	5	6
F	3	6	3
G	5	6	5
H	2	7	2
I	5	7	2
J	2	8	2
K	3	8	5
L	4	8	6
M	7	8	9
N	7	9	3
O	4	9	3
P	1	9	1
Q	8	9	8
R	9	10	1
S	5	11	2
T	8	11	2
U	11	12	3
V	6	12	2
W	10	12	3

Desenhando a rede e executando o algoritmo acima, chegaremos aos valores de

evento	cedo	tarde	C.Crit
1	0	0	*
2	7	7	*
3	9	19	
4	11	18	
5	13	13	*
6	18	34	
7	15	15	*
8	24	24	*
9	32	32	*
10	33	33	*
11	26	33	
12	36	36	*

Conclui-se portanto que a duração do projeto é de 36 unidades e que o caminho mínimo tem 8 eventos, a saber: o início (1), 2, 5, 7, 8, 9, 10 e o final (12).

☞ Para você fazer

1. Suponha a seguinte rede

tarefa	origem	destino	duração
A	1	2	9
B	1	3	6
C	3	4	8
D	1	4	3
E	3	5	4
F	1	5	9
G	2	5	8
H	2	6	7
I	1	7	8
J	4	8	5
K	5	8	5
L	8	9	7
M	5	10	4
N	8	10	3
O	9	11	2
P	9	12	8
Q	4	13	7
R	12	13	7
S	11	14	2
T	10	14	2
U	7	14	7
V	6	14	8
W	13	14	9

   1. Desenhe o grafo (a rede) de precedências
   2. Calcule os tempos mais cedo
   3. Calcule os tempos mais tarde
   4. assinale o caminho crítico (passa pelos nodos onde cedo=tarde)
   5. Responda: qual a duração do projeto e quantos nodos tem o CC (caminho crítico) ?

duracao	*nodos cc*

.

2. Suponha a seguinte rede

tarefa	origem	destino	duração
A	1	2	6
B	2	3	1
C	1	4	6
D	2	5	4
E	3	5	8
F	1	6	4
G	3	6	4
H	2	6	2
I	4	6	2
J	5	7	5
K	3	8	3
L	5	8	5
M	6	8	7
N	7	8	8
O	3	9	4
P	9	10	8
Q	8	11	9
R	9	12	5
S	10	12	2
T	11	12	8

   1. Desenhe o grafo (a rede) de precedências
   2. Calcule os tempos mais cedo
   3. Calcule os tempos mais tarde
   4. assinale o caminho crítico (passa pelos nodos onde cedo=tarde)
   5. Responda: qual a duração do projeto e quantos nodos tem o CC (caminho crítico) ?

duracao	*nodos cc*

Para saber mais: Planejamento com PERT-CPM, Henrique Hirschfeld, Editora Atlas.

azul - Sexta Brilhante

Descrição - Um problema complexo e grande. É impossível resolvê-lo à mão, há que lançar mão do computador. Baseado num problema modelo Euler (projecteuler.net), mas ligeiramente modificado, para que seja inédito. [1] Por exemplo, o Euler pode pedir o milhonésimo primo. Aqui pede-se o milhonésimo quinto primo.

Resposta esperada - Um único número.

Correção - Simples e rápida. Pode ser até on-line, mas neste caso há que se programar a infraestrutura de apoio. Isto foi feito na Universidade Positivo, no projeto que foi conhecido como Sexta Brilhante.

Infraestrutura Um sistema que identifique alunos e gestores, aceite respostas (conferindo-as) e atribua pontos por resposta correta a cada aluno. Facilidade de desafios e grupos de colegas. Criação de rankings. Um modelo excelente é o project euler.

A origem deste modelo de exercício é o maravilhoso site projecteuler.net. Aqui, problemas que exigem programação e matemática são apresentados e pede-se ao interessado que ache a resposta, que é composta por um único número. Este site foi prejudicado pela publicação em inúmeros lugares na Internet, das respostas aos problemas. O site pede explicitamente que as pessoas não façam isso, mas obviamente não foi obedecido. A mudança aqui foi sutil. O problema foi ligeiramente modificado para que o resultado não esteja publicado. Mais ainda, cada aluno recebe uma modificação diferente, com o mesmo objetivo. Veja-se por exemplo o problema:

```
Desafio do dia 12/04/13

Sequência mais longa: A seguinte sequência interativa  é definida para todos os inteiros positivos:

n → n/2(quando n  é par) e n → 3n + 1(quando n  é impar).
Usando esta regra e começando com 13, gera-se a seguinte sequência
13 → 40 → 20 → 10 → 5 → 16 → 8 → 4 → 2 → 1

Pode-se ver que esta sequência (começando em 13 e terminando em 1, contém 10 termos.
Ainda que não tenha sido provado, espera-se que todas as sequências terminem em 1.
Qual o número inicial, abaixo de N, produz a maior sequência em comprimento?
Note que depois que a cadeia começa, os termos podem passar de N.

N = 800000
```

Finalmente, nada impede que esta estratégia seja perseguida diretamente sobre o site projecteuler, apenas tendo em vista neste caso, que as respostas podem estar já publicadas e que portanto, eventuais desempenhos individuais precisam ser eventualmente relativizados.

laranja - Maratona

Descrição - Este é o ambiente padrão de maratona: um software capaz de receber programas enviados por alunos, munido de cargas inéditas de teste por problema e capaz de comparar a resposta produzida versus a esperada.

Resposta esperada - Resposta binária: acertou ou errou.

Correção - Automática

Infraestrutura Um ambiente completo de maratona, funcional e operante 24/7. Há duas dificuldades e uma complicação: as dificuldades são a operação do servidor, com todo o trabalho de manutenção preventiva e corretiva. A segunda dificuldade é a criação, correção, validação, depuração e manutenção do acervo de problemas. A complicação é representada pela tentação (perfeitamente válida e talvez até recomendável) de usar um servidor mundial já existente. Uma vantagem residual, mas importante é preparar o público alvo para competições exógenas. Um aspecto disto é que tais problemas em sites mundiais estão em inglês, como aliás são todas as competições internacionais.

[1] Infelizmente o grande apelo positivo da proposta original do projeto Euler acabou se desvirtuando: hoje há dezenas de sites na Internet que publicam os resultados numéricos de boa parte dos problemas propostos no Euler: ficam de fora os problemas realmente complexos, mas que por isso mesmo se encontram fora do alcance da grande maioria dos usuários. Os que efetivamente poderiam melhorar as habilidades de programação da humanidade têm seus resultados esperados publicados. De ponto de vista individual, não há grande problema: basta que o interessado se abstenha de procurar tal resultado. Mas, do ponto de vista institucional, perde-se muito: não é possível avaliar um aluno pela posse do resultado, já que não se sabe como ele o conseguiu.

Pode-se usar o excelente site https://uva.onlinejudge.org/, que aliás conta com abundantes alternativas de linguagens:

- ANSI C 5.3.0 - GNU C Compiler
- JAVA 1.8.0 - OpenJDK Java
- C++ 5.3.0 - GNU C++ Compiler
- PASCAL 3.0.0 - Free Pascal Compiler
- C++11 5.3.0 - GNU C++ Compiler
- PYTH3 3.5.1 - Python 3

Lembrando que cada alternativa de linguagem distinta implica em trabalho significativo de apoio e garantia de funcionamento.

roxo - 3 alternativas

Descrição - É dado um problema (na verdade, uma lista de problemas) e são dadas 3 respostas, sendo duas erradas e uma certa. Cabe ao aluno identificá-la. Novamente, cada aluno tem instâncias únicas e não consegue compartilhar respostas. Os problemas tendem a ser mais simples e eventualmente podem ser resolvidos via lápis e papel.

Resposta esperada - Um conjunto de respostas de múltipla escolha.

Correção - Fácil, como são fáceis as correções de múltipla escolha.

Um exemplo é o exercício VIVOp65.

U Positivo - UTFPR - PUCPr – 23/01/2019 - 10:39:05.5
Prof Dr P Kantek (pkantek@gmail.com)
Prática VIVOp65a, V: 1.08 1

- 1 _____ / _____ / _____

Problemas

1. Supermercado Imagine uma cidade toda quadriculada, com ruas (leste-oeste) e avenidas (norte-sul). Aqui, os endereços sempre estão na forma (x, y), onde x é o número da rua e y é o número da avenida. Seu endereço é dado, na forma de um par de números, isto significa que você mora em uma esquina. Também são dados os endereços de 3 supermercados, também em esquinas, identificados como A, B e C. Você deve calcular a distância pitagórica (ou seja em linha reta) desde sua casa até cada mercado e responder qual dos 3 mercados fica mais próximo (em linha reta) da sua casa. Por exemplo, se você mora em (29,3) e os mercados ficam em A=(7,9); B=(17,29) e C=(20,14), o mais próximo é o C.

2. Ainda o mercado A definição deste exercício 2 é muito parecida ao exercício 1, mas com uma diferença importante. A distância agora nao é mais em linha reta, mas considerando um trajeto de carro (ou mesmo a pé) convencional entre os 2 pontos. Esta distância também é conhecida como distância Manhattan. Agora, dados seu novo endereço e os 3 novos mercados (A, B e C) você deve dizer qual o mercado mais próximo. Por exemplo, se você mora em (3,23) e os mercados ficam em A=(12,27), B=(17,28) e C=(15,9), o mais próximo é o A.

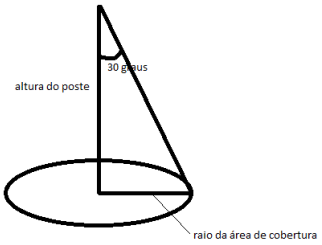
3. Investimento Você tem uma certa quantia em dinheiro e você não vai precisar usá-la pelos próximos 6 meses. Você sabe que dinheiro parado cria mofo e é devorado pela inflação. Assim, você vai investí-lo. O banco A remunera o capital investido a uma dada taxa de juros (compostos) mensal. O banco A cobra uma quantia fixa para fazer o negócio. O banco B também tem a sua taxa de juros, e ele nao cobra nenhuma quantia. O banco C aplica sua taxa e ainda te dá um brinde (mimo) em dinheiro se você fizer negócio com ele. Dados o capital, as 3 taxas de juros e os valores da cobrança e do mimo, descubra qual das 3 alternativas é mais interessante para você. Exemplo: se o valor é 110.000, as taxas dos bancos são: A=0,91% ao mês, B=0,65% e C=0,52% a cobrança é de R$ 1697,98 e o mimo é de R$ 783,39, a melhor alternativa é a A.

4. Média Uma certa disciplina calcula suas médias fazendo a seguinte conta: Cada exercício é pontuado de 0 a 10. Existem 6 exercícios no bimestre. A média dos exercícios entra com peso 6 na média e a nota da prova entra com peso 4 na média. Dados 3 alunos A, B e C com suas respectivas notas de exercício e prova pede-se qual o aluno que teve maior média. Por exemplo, se as notas foram:

al	pr	f1	f2	f3	f4	f5	f6
A	10	2	1	2.5	9.5	5.5	4.5
B	9	8	6	2.5	5.5	8.5	8
C	10	5	6.5	0.5	10	9.5	2.5

O aluno de melhor média é o B.

5. Detetor de gatos Um grande cientista inventou um detector de gatos. Porque alguém precisaria saber quando um gato aparece no pedaço é uma questão em aberto, mas esse não é o ponto aqui. O detector precisa ser instalado no alto de um poste e portanto se precisa considerar 2 custos: o do detector e do poste. O custo do poste depende da altura do mesmo e tem a seguinte lei de formação: até 10m, o poste custa x R$ por metro. Acima de 10 m, além do custo anterior há ainda um custo de $y^{1.2}$ para cada y metros acima dos 10 m originais. O detector colocado sobre o poste sempre têm o mesmo preço, portanto ele pode ser desconsiderado. O detector tem uma área de abrangência que depende da altura onde ele é colocado (quanto mais alto, maior a abrangência). Para calcular a área de cobertura, considere um cone cuja abertura angular na extremidade é de 30^o para cada lado. Veja-se na figura, como a coisa funciona:



Existirão 3 possibilidades de construir o detector de gatos. Cada uma delas terá uma altura específica e você deve descobrir qual das 3 garante o menor custo por metro quadrado de cobertura. Por exemplo: Se o poste de A tem 17 m, B tem 18m e C tem 13m, as áreas de cobertura são 302.64, 339.29 e 176.97 e os custos de cada poste são $P_1 : x = 139, y = 105$, $P_2 : x = 149, y = 111$ e $P_3 : x = 115, y = 141$, o que vai gerar custos unitários por m^2, de 10.75, 11.10 e 12.92. Assim o mais em conta é o A.

6. Bananas Uma empresa A vende bananas por um determinado preço por dúzia. A empresa B vende bananas por catorzena e a empresa C vende por preço unitário de bananas. Supondo que você quer comprar uma quantidade fixa de bananas, qual empresa oferece o melhor preço ? Por exemplo, se A cobra 3.04 R$/dúzia, B 3.86 R$/catorzena e 0.29 R$ por unidade, para comprar 10000 bananas, a empresa melhor é a A, cujo custo unitário é 0.25/banana. O custo de B é 0.27/banana e o C está dado no problema.

7. Suco Você quer fazer suco de frutas. Existem 3 alternativas. A primeira (A) é comprar as frutas. Neste caso há um custo a cada certa quantidade de fruta e para fazer 1 litro de suco gastam-se 500 gr de fruta. A alternativa B é usar um xarope concentrado que custa um certo preço e faz uma certa quantidade de suco. Finalmente a alternativa C usa suco em pó, que precisa ser rehidratado. Há um custo por grama de suco em pó e também uma especificaçao de quanto pó é necessário para 1 l de suco. Se o objetivo é fazer uma certa quantidade de litros, e supondo que não há custo associado ao trabalho de preparo e que a qualidade do resultado obtido é a mesma nos 3 casos, qual a mais barata ? Por exemplo, se 200 gr de fruta custam 2,43 R$, o xarope custa 52,20 por litro e deve ser usado na proporçao 1:9 (ou seja 0.111 litros de xarope fornecem 1 l de suco) e o pó custa 0,163 R$ por grama e são necessários 43 gramas para fazer 1 litro, a alternativa mais em conta é a B. Observação importante: na alternativa 2 desconsidere o volume do suco. Considere apenas o volume de água usada.

8. Carro Um carro triflex (gasolina, alcool e eletricidade) tem suas medidas de desempenho em cada um dos modais de energia. Dados seus custos individuais, pede-se qual o modal mais barato. Desconsidere questões sobre desempenho, comodidade, disponibilidade, etc. Exemplo: seja o carro faisca que pode usar os 3 modais com os desempenhos: A=gasolina, faz 11.86 km/l. B=alcool, faz 8 km/l e C=eletricidade faz 3,21 km/Kw. Os preços são: 3,60 R$/l de gasolina 1.78 R$/l de alcool e 1.36 R$/Kw a energia elétrica. Neste caso, a alternativa mais barata é a B.

9. Pendrive Existem no mercado diversas opções de armazenamento de dados. Suponha que você tem a necessidade de armazenar 7.5GB de dados (8.053.063.680 bytes) e para você é indiferente quebrar ou não o arquivo, assim como não lhe interessa ter um eventual espaço sobrando. Assim, dados os preços dos pendrives de 1, 2 e 16GB qual a alternativa mais barata para você ? Por exemplo, supondo que (A) um pendrive de 2GB custa R$ 12,60, (B) um de 1GB R$ 7,10 e (C) um de 16GB custa R$ 43,20, a alternativa mais barata é a alternativa C.

10. Impressão de um trabalho Você precisa imprimir 54 cópias de um trabalho que compôs em seu computador. As alternativas de que dispõe são 3: A=imprimir as 54 cópias diretamente em sua impressora, B=imprimir uma cópia do trabalho na impressora e tirar mais 53 num copista perto de sua casa. Finalmente, a alternativa C é levar o arquivo num pendrive até uma gráfica e lá encomendar a impressão dos 54 volumes. Dados todos os preços envolvidos, você deve achar a alternativa mais barata. Desconsidere o trabalho desigual que terá nas 3 alternativas, considere apenas o custo monetário. Por exemplo, se o seu trabalho tem 67 páginas, e o custo de uma cópia em sua casa é de 0,056 R$/cópia, se o copista cobra 0,044 R$/cópia e se a gráfica cobra 4,556 R$/volume, a alternativa mais em conta é a B.

☞ Para você fazer

1. Supermercado-Pitágoras A sequência de números é x, y de sua casa, e x, y dos 3 mercados, A,B e C na ordem:

```
30  9   14  9   22 21    6 11
```

2. Supermercado-Manhattan A sequência de números é x, y de sua casa, e x, y dos 3 mercados, A,B e C na ordem:

```
24  3    1 10   12 28   16   18
```

3. Investimento A sequência de números: valor a investir, a taxa de juros mensais dos bancos A,B e C, o valor cobrado pelo banco A e o presente dado pelo banco C.

```
104000 1.0077 1.0075 1.0051  229.62 1650.38
```

4. Média Aqui são 3 linhas. Em cada linha, o primeiro número é a nota da prova e os outros 6 números são as notas individuais de cada folha.

```
 7.5 10.0  2.0 10.0  9.5  4.5  7.0
 8.0  6.0   .5  2.0  5.5  1.5  5.5
 9.0  8.5  5.5  5.0  5.5  2.0  1.0
```

5. Detetor de gatos Há 3 conjuntos de dados (A, B e C). Em cada um, há a altura do poste e os parâmetros x, y para cálculo do custo do poste

```
17 104 124      12 117 122      13 138 129
```

6. Bananas Há 3 valores: o preço da dúzia (A), o da catorzena (B) e o da banana (C)

```
3.14   4.00    .25
```

7. Suco Há 6 valores: m, n, p, q, r, s. Os dois primeiros: n gramas de fruta custam m reais. O litro de concentrado custa p reais e para 1 litro de suco usa-se $1/q$ concentrado. Finalmente, o grama de pó custa r reais e para 1 litro de suco usam-se s gramas de pó.

```
2.30   200  72.96 12  .141  50
```

8. Carros Há 6 valores: os 3 primeiros são os preços de gasolina, etanol e eletricidade. Os últimos 3 são a quantidade de quilometros que o carro faz com 1 litro de gasolina, etanol e 1 Kw de energia elétrica.

```
3.87   1.23   1.32  11.43   6.14   3.21
```

9. Pendrive Há 3 preços: o de 2GB, o de 1GB e o de 16GB.

```
12.40   7.40  43.20
```

10. Impressão do relatório Há 4 valores: o primeiro é o custo da página impressa na sua casa. O segundo é o preço da cópia no copista e o terceiro é o preço de um exemplar completo no birô de impressão. O quarto número é o número de páginas do relatório. Lembre que sempre são 54 cópias do relatório.

```
.056    .048   6.039   99
```

Escreva A, B ou C no quadro abaixo, para cada exercício

1	2	3	4	5	6	7	8	9	10

marrom - Arquivão de entrada

Descrição - Uma série de dados para diversas instâncias de um único problema. Esta quantidade inviabiliza a solução via papel e lápis, há que se usar o computador.

Resposta esperada - As respostas de todas as instâncias são somadas (eventualmente usando-se a função módulo em algum momento da soma: com isso de impede o *overflow* do resultado

Correção - Agindo-se assim, a correção é simples: basta conferir um único número.

Eis alguns exemplos: VIVO055, no qual 50 tabuleiros preenchidos de xadrez são entregues ao aluno e ele deve efetuar calculos posicionais sobre tais tabuleiros. Ainda no mesmo exercício são apresentadas 50 operações aritméticas ad-hoc e o aluno deve calcular 50 operações novas. Depois 20 matrizes distintas são apresentadas semi-vazias, devendo o aluno maximizar seu preenchimento e finalmente 20 sequências de Fibonacci devem ser calculadas. Como todos os valores são somados, a correção se limita a verificar 4 números.

1. Posições no xadrez

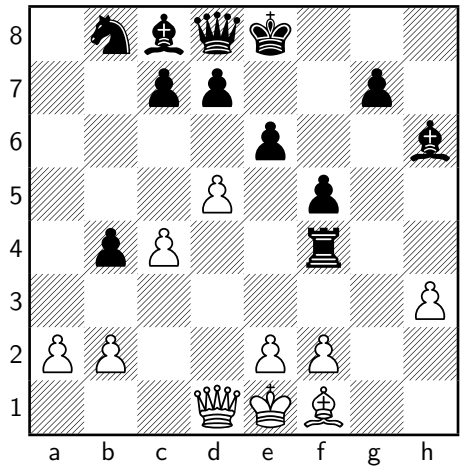
Imagine que você precisa examinar um tabuleiro de xadrez. Trata-se de uma matriz numérica de 8 × 8, com casa brancas e negras alternadas sendo que que a casa embaixo, mais à direita, é branca. Por ser numérica, a matriz usa números para identificar as peças. Primeiro, a cor. Peças brancas são positivas e negras, negativas. Os valores: 1=peão, 2=torre, 3=cavalo, 4=bispo, 5=dama e 6=rei. Você deve escrever um programa que vai receber um tabuleiro preenchido com uma situação de meio-jogo e também uma linha (1-8) e coluna (1-8). Você deve olhar o tabuleiro e se essa posição estiver zerada, deve responder zero. Se ela contiver uma peça, deve examinar:

- quantas casas essa peça pode ocupar na próxima jogada
- quais peças contrárias podem ser tomadas na próxima jogada. Para este cálculo, some o valor da peça a ser tomada em valor absoluto.

Finalmente, não se assuste se aparecer um tabuleiro com alguma situação muito infrequente do ponto de vista enxadrístico: o gerador usa e abusa de números aleatórios. Por exemplo: No tabuleiro

```
     1   2   3   4   5   6   7   8
  +--------------------------------
1|   0  -3  -4  -5  -6   0   0   0
2|   0   0  -1  -1   0   0  -1   0
3|   0   0   0   0  -1   0   0  -4
4|   0   0   0   1   0  -1   0   0
5|   0  -1   1   0   0  -2   0   0
6|   0   0   0   0   0   0   0   1
7|   1   1   0   0   1   1   0   0
8|   0   0   0   5   6   4   0   0
```

que corresponde a esta posição



Se o endereço a pesquisar fosse 5,6 a resposta deveria ser 7,2
Se fosse 2,3 a resposta deveria ser 2,0
Se fosse 1,2 a resposta deveria ser 2,0

Seu programa deverá ler 50 tabuleiros no arquivo e somar os 2 valores acima para os 50 tabuleiros. A distribuição de dados no arquivo é a seguinte: A primeira linha tem L, C seguido de 6 números 999. Tanto L quanto C tem origem dos índices = 1. Na sequência, vem o tabuleiro (8 linhas de 8 colunas), positivos são as brancas, negativos, as negras e 1=peão, 2=torre, 3=cavalo, 4=bispo, 5=dama e 6=rei. Há um total de 450 linhas aqui.

2. Operações mandrake

A seguir, uma lista de operações matemáticas simples (identificadas por letras maiúsculas). Você precisa descobrir o que cada função faz e depois deve ler 50 expressões no arquivo e achar a soma das respostas a elas. Acompanhe os exemplos:

```
2 A 4 = 10
3 A 8 = 27
4 A 27 = 112
5 A 1 = 10
```

Neste exemplo, a função x A y multiplica x por y e soma x ao resultado. A seguir exemplos das 4 funções que você vai encontrar no seu arquivo

```
9 A 9 = 90          6 C 6 = 72          3 D 3 = 18
9 D 2 = 22          5 B 6 = 36          5 D 2 = 14
7 B 5 = 40          8 B 6 = 54          9 D 1 = 10
6 D 3 = 27          9 B 5 = 50          2 C 7 = 18
3 B 4 = 16          4 A 8 = 36          6 D 8 = 112
4 D 1 = 5           4 C 9 = 52          8 D 8 = 128
7 D 3 = 30          1 C 7 = 8           9 A 5 = 54
3 A 6 = 21          1 B 5 = 10          1 A 7 = 8
9 C 3 = 108         1 D 2 = 6           2 C 3 = 10
5 D 8 = 104         3 C 4 = 21          9 A 6 = 63
3 A 7 = 24          3 A 5 = 18          4 A 4 = 20
8 C 1 = 72          8 C 9 = 136         6 C 4 = 60
6 A 2 = 18          1 B 4 = 8           8 C 1 = 72
3 D 2 = 10          1 A 9 = 10          3 C 1 = 12
2 D 9 = 99          5 C 4 = 45          6 D 8 = 112
```

Leia no arquivo uma série de 50 operações e calcule a soma de todos os resultados.

3. Maior valor

Você deve terminar de preencher uma tabela 5 × 5 que já tem algumas casas preenchidas. Duas casas são consideradas vizinhas se têm um vértice ou um lado em comum. As regras que você precisa respeitar são:

- um casa só pode ser preenchida se alguma de suas casas vizinhas já contiver algum número.
- ao preencher uma casa, deve-se colocar a soma de todos os números que já constam em suas casas vizinhas.

A pergunta a responder é: qual é o maior número que é possível escrever na tabela ? Veja a seguir um exemplo completo:

```
0 0 0 0 0            214  211 207 167 333
0 3 1 0 0            428    3   1  36 130
0 0 0 0 0    --->    885  454   8  27  67
0 0 0 4 0           2702 1363  14   4  98
0 0 2 0 0           8146 4081   2 118 220
```

cuja resposta é 8146.

Leia no arquivo um conjunto de 20 matrizes 5 × 5 e some os 20 maiores números (um para cada matriz). Note que há uma linha contendo 999 que deve ser ignorada (é apenas uma separação visual).

4. Números de Fibonacci

É conhecida a sequência de Fibonacci: $1, 1, 2, 3, 5, 8, 13, 21, 34, \ldots$ na qual $a_n = a_{n-1} + a_{n-2}$. No exercício a seguir, você deve ler no arquivo 20 quádruplas de números, a saber, a, b, lim_i e lim_s. Com os dois primeiros, deve construir uma sequência de fibonacci e a seguir deve contar quantos números dessa sequência estão incluídos entre os limites dados a seguir. Todos os números são inferiores a 110.000.000. Eis alguns exemplos

```
12   16          3097062          26253069   4
26   45          8849945          13274179   1
17   37          1349137          52320948   8
37   51          1083061          32957279   7
24   50          2196396          51256339   6
```

☞ Para você fazer

Leia o seu arquivo publicado no lugar de sempre com o nome de

`F0550001.myd`

Estão no mesmo lugar os arquivos F055EXE1.MYD, F055EXE2.MYD e F055EXE3.MYD cujas respostas são:

```
142   56   2027    1076816    105
122   25   2072    1367135    119
 90   42   1953    1896456     96
```

1.1 casas	1.2 peças	2. 50 operações	3. 20 maiores	4.fibo

vermelho - Algoritmo incompleto

Descrição - Dado um problema, o algoritmo em Python é apresentado com algumas lacunas. Cada aluno tem lacunas distintas, e há lacunas que surgem em todos os exercícios de todos os alunos.

Resposta esperada - Uma instância numérica ou alfanumérica simples que deve ser obtida ao aplicar o algoritmo (corretamente completado) aos dados de entrada individuais do aluno.

Correção - Basta comparar o resultado obtido vis-a-vis o gabarito prévio.

Um exemplo desta classe é o VIVO146, no qual o algoritmo de Huffman é apresentado. Dada uma instância de compressão o aluno deve achar a configuração binária do caráter espaço (␣)

Comprimindo objetos binários com Huffman

O algoritmo de Huffman foi proposto em 1952, (*A Method for the Construction of Minimum-Redundancy Codes*) e é um marco na compressão de dados. Necessário não apenas pela quantidade imensa de dados digitais gerados pela humanidade, como pelo fato de que eles tendem a ser muito redundantes (música, vídeo, imagens, mesmo texto). Veja uma conta simples: Uma vídeo HDTV tem 1920×1080 pixels, e uma nova imagem é gerada a cada 1/30 de segundo e cada pixel ocupa 3 bytes (RGB), ou seja, sem compressão, um vídeo HDTV ocupa 186.624.000 bytes/segundo. 1 hora, 671.846.400.000 bytes, ou o conteúdo de 133 DVDs. Que você possa ver um filme de mais de 1 hora em um único DVD deve-se à compressão.

Dados não comprimidos usam uma tabela de possíveis valores existentes no material e atribuem a cada valor uma certa configuração de bits. Por exemplo, no código ASCII, a letra A maiúscula tem configuração 0100.0001. A letra Z é 0101.1010. Ambas ocupam 8 bits. O que Huffman intuiu é que a letra A é muito mais frequente do que a letra Z, e que, se ela tivesse um comprimento menor, haveria um ganho considerável de espaço. Vale aqui a regra de Pareto: em qualquer objeto binário poucos valores ocupam a maior parte das ocorrências. Isto caracteriza a redundância.

Como não existe almoço grátis, o que se ganha em espaço se perde em facilidade: agora não existe um comprimento padrão em bits (8 no caso do código ASCII, mais de 20 no caso de Unicode), e cada caracter tem um comprimento diferente. Os mais frequentes são mais curtos que os menos frequentes. O que Huffman fez foi bolar um algoritmo que trata este fato e permite facilmente comprimir e descomprimir objetos digitais.

A idéia de Huffman é:

1. Contar os caracteres e ordená-los
2. Construir uma árvore binária
3. Construir um dicionário a partir da árvore
4. Percorrer a árvore (ou consultar o dicionário) na compressão
5. Percorrê-la em sentido inverso na descompressão

1.Contagem

Eis o algoritmo

```
def huf2(uni):
  ctd=[]
  for i in range(len(uni)):
    j=0
------------------------------
      if uni[i]==ctd[j][0]:
        ctd[j][1]=ctd[j][1]+1
        break
      j=j+1
------------------------------
      ctd.append([uni[i],1])
  cts=sorted(ctd,key=getkey)
  return cts
```

2. Montagem da Árvore

Eis o algoritmo

```
def ca1(x):
  for j in range(len(x)):
    x[j][0]=-ord(x[j][0])
  arv=[]
  i=1
--------------------------------
    x=sorted(x,key=getkey)
    w=x[0][1]+x[1][1]
    alfa=x[1][0]
    if alfa>=0:
      alfa=alfa-1
    beta=x[0][0]
    if beta>=0:
       beta=beta-1
    arv.append([i-1,alfa,beta])
    x.append([i,w])
    x.pop(0)
--------------------------------
    i=i+1
  return arv
```

3. Montagem do dicionário

Com a árvore acima, monta-se o dicionário

```
global dicionario
dicionario=[]
cdic1(len(bb)-1,'',bb) # bb é a arvore
...
def cdic1(raiz,ateaqui,arv):
  if arv[raiz][1]<0:
-----------------------------------------------
    dicionario.append([chr(-arv[raiz][1]),xica])
  else:
    cdic1(arv[raiz][1],ateaqui+'0',arv)
-----------------------------------------------
    xica=ateaqui+"1"
    dicionario.append([chr(-arv[raiz][2]),xica])
  else:
    cdic1(arv[raiz][2],ateaqui+'1',arv)
```

4. Compressão

Eis o algoritmo

```
def comprime1(texto,dicionario):
-------------------------------------
  for i in range(len(texto)):
    for j in range(len(dicionario)):
-------------------------------------
        res=res+dicionario[j][1]
        break
  return res
```

5. Descompressão

```
def descomprime(tc,arvore):
  td=''
  raiz=len(arvore)-1
  i=0
  while i<len(tc):
    if tc[i]=='0':
      raiz=arvore[raiz][1]
----------------------------
        td=td+chr(-raiz)
        raiz=len(arvore)-1
    else:
---------------------------
      if raiz<0:
        td=td+chr(-raiz)
        raiz=len(arvore)-1
    i=i+1
  return(td)
```

6. Para gravar

```
def gravac(texto,onde):
  f=open(onde,"wb") ; tt=''
-------------------------------------------------
  tamu=8*(math.ceil(tamr/8))
  a1=tamr%256
  tamr=tamr//256
  a2=tamr%256
  tamr=tamr//256
  a3=tamr%256
  tamr=tamr//256
-------------------------------------------------
  texto=texto+'0000000'
  tt=bytes([a4])+bytes([a3])+bytes([a2])+bytes([a1])
  i=0
  while i<tamu:
    tt=tt+bytes([int(texto[i+0:i+8],2)])
    i=i+8
  f.write(tt)
  f.close()
```

7. Para ler

```
def lec(onde):
  f=open(onde,"rb")
  t=f.read()
  tam=(t[0]*16777216)+(t[1]*65536)+(t[2]*256)+t[3]
  tt=''
-------------------------------------------------
  while i<(len(t)):
    tt=tt+bin(t[i])[2:].rjust(8,'0')
-------------------------------------------------
  return tt[0:tam]
```

O conjunto

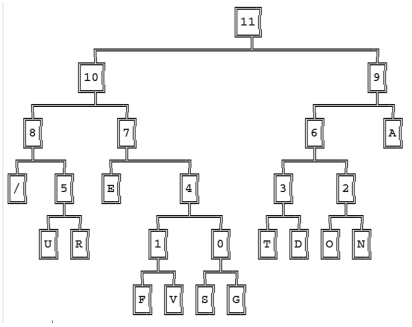
```
aa=huf2('IVO/VIU/A/UVA')
bb=ca1(aa)
print('Lista de caracteres')
print(aa)
print('Arvore de Huffman')
print(bb)
global dicionario
dicionario=[]
cdic1(len(bb)-1,'',bb)
print('Dicionário')
print(dicionario)
tc=comprime1('IVO/VIU/A/UVA',dicionario)
print('Resultado da compressão')
print(tc)
td=descomprime(tc,bb)
gravac(tc,'c:/p/n/451/arquivo1')
lido=lec('c:/p/n/451/arquivo1')
td=descomprime(lido,bb)
print('Resultado da descompressão')
print(td)
```

Exemplos

Seja a frase:

`E/NAO/DE/AGRESTE/AVENA/OU/FRAUTA/RUDA - / é 000`

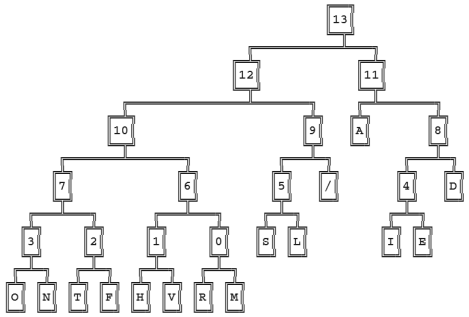
A árvore gerada é



Mais um exemplo:

`MARAVILHA/FATAL/DA/NOSSA/IDADE - / é 011`

Neste último exemplo, a árvore é



☞ Para você fazer

Suponha que – usando os algoritmos acima – precisa comprimir a frase a seguir, extraída d'Os Lusíadas:

`SE/A/TANTO/ME/AJUDAR/O/ENGENHO/E/ARTE`

Qual será a configuração do espaço (caracter /) ?