COMPETIÇÃO DE PROGRAMAÇÃO

9 E 10 DE MARÇO – 2019

IEEE
University of Aveiro Student Branch

NEI

CAPÍTULO I – DESAFIOS DE NÍVEL FÁCIL

Desafio Nº: 1

Pontos: 5

Título: Número de pares cuja soma é divisível por 5.

Descrição do problema: É dada uma lista de inteiros positivos com N elementos. O objetivo é calcular o número de pares de inteiros na lista cuja soma divisível por 5.

Considerações: Relativamente à primeira linha do input terá de ser um número inteiro positivo. Já o segundo terá os elementos da lista separados por um espaço.
O output será o número de pares dessa lista cuja soma é divisível por 5.

Linguagem preferencial:Java, Python

Exemplo:
Input: *Output: 3*
6
5 1 8 3 2 9

Desafio Nº: 2 - corrigido

Pontos: 5

Título: Contador de divisores

Descrição do problema: É dado dois números inteiros, A e B. O objetivo é contar quantos números no intervalo [A, B] têm um número par de divisores.

Considerações: A linha de input conterá os números A (igual ou maior que 1) e B (menor ou igual a 100, separados por um espaço. O output será o número de valores com um número par de divisores.

Linguagem preferencial:Java, Python

Exemplo:
Input: *Output: 7*
1 10

Desafio Nº: 3 – corrigido o output

Pontos: 5

Título: Letra

Descrição do problema: É dada uma lista de N palavras. Será guardada de cada palavra a letra com menor valor alfabético (a<b). O objetivo é retornar uma *string* com o menor valor alfabético com as letras obtidas anteriormente.

Considerações: A primeira linha de input será o número de palavras da lista. A segunda linha conterá as N *strings* separadas por espaço, representando uma das palavras. O output será uma *string* do tamanho N.

Linguagem preferencial:Java, Python

Exemplo:
Input: *Output: adg*
3
gol abc tiroide

Desafio Nº: 4

Pontos: 5

Título: Diferença entre extremos

Descrição do problema: É dada uma lista de 6 números. O objetivo é retornar a máxima amplitude entre quaisquer dois inteiros da lista.

Considerações: O input será 6 números inteiros separados por espaço. O output é o valor retornado da função.

Linguagem preferencial: Java, Python

Exemplo:
Input: *Output: 13*
1 3 9 -4 5 -1

CAPÍTULO II – DESAFIOS DE NÍVEL MÉDIO

Desafio Nº: 5

Pontos: 10

Título: Balança

Descrição do problema: É dada uma lista de 2 números inteiros positivos e outra lista com vários números inteiros positivos. A primeira lista contém dois pesos, que corresponde a cada lado da balança (lado esquerdo para o primeiro elemento e lado direito para o segundo elemento da primeira lista), e a segunda lista são os pesos possíveis de serem usados para equilibrar o peso de cada lado, sendo que só pode ser usado no máximo desta lista dois pesos. O objetivo é retornar os pesos usados da segunda lista para equilibrar indicando também de que lado é que cada um foi colocado.

Considerações: A primeira linha de input será os dois elementos da primeira lista separados por espaço, tendo que estes serem diferentes. A segunda linha de input será com os pesos de balanceamento, sendo que terá de ser uma lista de pelo menos 3 elementos. O output será os valores obtidos para equilibrar a balança. Caso não haja solução o programa deverá retornar que a sequência não é possível.

Linguagem preferencial: Java, Python

Exemplos:

Input1:
1 3
4 9 6 1

Output1: "4 – lado direito, 6 – lado esquerdo"

Input2:
1 4
4 8 6 2

Output2: "Não é possível balancear"

Input3:
1 2
4 8 6 1

Output3: "1 – lado esquerdo"

Desafio Nº: 6 – corrigido

Pontos: 10

Título: Caminho num bloco

Descrição do problema: É dada uma *string* que representa os movimentos feitos num bloco 4x4 células iniciando do canto superior esquerdo. Os carateres da *string* serão os seguintes: d – direita, e – esquerda, c – cima, b – baixo, ? – desconhecido. O objetivo é descobrir que movimentos são os pontos de interrogação de forma para que o caminho seja criado para ir do canto superior esquerdo do bloco até ao canto inferior direito sem repetir células anteriormente traçadas no bloco.

Considerações: A linha de input será uma *string* com pelo menos dois pontos de interrogação. O output será a(s) *string(s)* com a substituição dos pontos de interrogação pelo movimento correto.

Linguagem preferencial: Java, Python

Exemplos:

Input: *Output:* *bbbddcdb*

???ddc?b

Desafio Nº: 7

Pontos: 10

Título: Anagrama

Descrição do problema: É dada uma lista com N palavras. O objetivo é retornar a lista com mais palavras que são anagramas.

Considerações: A primeira linha de input será o número de palavras que irá conter a lista fornecida, que terá de ser maior que 2. A segunda linha de input conterá as N palavras separadas pelo espaço. O output será a lista obtida. Caso não haja nenhum valor retornado deverá ser indicado que não foi encontrado anagramas.
Um anagrama é uma espécie de jogo de palavras, resultando do rearranjo das letras de uma palavra ou expressão para produzir outras palavras ou expressões, utilizando todas as letras originais exatamente uma vez.

Linguagem preferencial: Java, Python

Exemplos:

Input1:
3
drone dreno cair

Output1: drone, dreno

Input2:
4
acabar jogo linha ferro

Output2: "Não existe anagramas."

Desafio Nº: 8 - corrigida

Pontos: 10

Título: Maior série de números numa lista

Descrição do problema: É dada uma lista com N números inteiros positivos não repetidos ordenada pela ordem crescente e um valor K representando o número de números inteiros que se pode colocar na lista inicial. O objetivo é colocar os K números de forma a ter na lista uma maior série de números consecutivos e retornar o tamanho dessa série.

Considerações: A primeira linha de input conterá os valores de N e K separados por um espaço.
A segunda linha linha de input conterá a lista.
O output será o tamanho da maior série números consecutivas da lista.

Linguagem preferencial:Java, Python

Exemplos:
Input: *Output: 8 (colocando os números 12, 13, 16 e 17)*
8 4
1 2 3 4 10 11 14 15

Desafio Nº: 9

Pontos: 10

Título: Soma de números Fibonacci

Descrição do problema: É dado um número inteiro positivo N maior que 2. O objetivo é obter N como uma soma de números Fibonacci com o menor número de termos possível e retorna a lista com esses termos.

Considerações: O input é o valor de N.
O output será os números da lista obtida separados por espaço.
Números de Fibonacci: 1,1,2,3,5,8,13…

Linguagem preferencial:Java, Python

Exemplos:

Input1: 8 *Output1: 3 5*

Input2: 14 *Output2: 1 13*

Desafio Nº: 10

Pontos: 10

Título: Menor número possível concatenado

Descrição do problema: É dado uma lista de N números inteiros positivos. O objetivo é retornar o menor número que se pode obter com a concatenação de todos os N números.

Considerações: A primeira linha do input é o valor de N. A segunda linha é a lista com N números inteiros positivos separados por espaço.
O output será o valor obtido da concatenação.

Linguagem preferencial:Java, Python

Exemplos:
Input: *Output: 1120305*
4
120 5 30 1

Desafio Nº: 11

Pontos: 10

Título: Contador de palíndromos

Descrição do problema: Um palíndromo é uma palavra, frase ou qualquer outra sequência de unidades que tenha a propriedade de poder ser lida tanto da direita para a esquerda como da esquerda para a direita. Por exemplo, mom e tacocat são palíndromos, assim como quaisquer cadeias de caracteres únicas.
Dada uma *string*, determine quantas *substrings* da *string* fornecida são palíndromos.

Considerações: O input é a string. O output será o valor retornado da função.

Linguagem preferencial: Java, Python

Exemplos:
Input: tacocat

Output: 10 (é o tamanho da lista [‘t’, ‘a’, ‘c’, ‘o’, ‘c’, ‘a’, ‘t’, ‘coc’, ‘acoca’, ‘tacocat’])

Desafio Nº: 12 – repetido ao desafio 10

Pontos: 10

Título: Menor número possível concatenado

Descrição do problema: É dado uma lista de N números inteiros positivos. O objetivo é retornar o menor número que se pode obter com a concatenação de todos os N números.

Considerações: A primeira linha do input é o valor de N. A segunda linha é a lista com N números inteiros positivos separados por espaço.
O output será o valor obtido da concatenação.

Linguagem preferencial:Java, Python

Exemplos:
Input: *Output: 1120305*
4
120 5 30 1

Desafio Nº: 13

Pontos: 10

Título: Divisores ímpares

Descrição do problema: É dado vários intervalos de inteiros [A, B]. Para cada número desse intervalo, calcule o seu maior divisor ímpar. O objetivo é retornar para cada intervalo a soma desses divisores obtidos.

Considerações: A primeira linha do input contém um inteiro N que representa o número de casos de teste que se seguirão. As linhas de input seguintes (N linhas) consistem em dois valores inteiros A e B separados por espaço.
As linhas de output serão as somas de cada intervalo.

Linguagem preferencial:Java, Python

Exemplos:

Input:	*Output:*
3	
1 5	*11 (1+1+3+1+5)*
2 4	*5 (1+3+1)*
3 7	*19 (3+1+5+3+7)*

Desafio Nº: 14

Pontos: 10

Título: Array Mex

Descrição do problema: É dada uma lista de N números inteiros. O objetivo é encontrar e retornar o menor número inteiro não negativo que não ocorre na lista. O problema terá de ser resolvido em O(N) usando como memória extra O(1).

Considerações: A primeira linha do input é o valor de N. A segunda linha é a lista com N números inteiros não negativos separados por espaço.
O output será o valor obtido da função.

Linguagem preferencial: Java, Python

Exemplo:

Input: *Output: 0*

4

8 3 2 7

Desafio Nº: 15

Pontos: 10

Título: Formatação de texto para justificado.

Descrição do problema: É dada uma lista de N palavras (*string)* reprentando um texto e um valor K que serve para formatar o texto em várias linhas de modo a que cada linha tenha o comprimento K e seja justificado. O objetivo é retornar o texto com o maior número de palavras possível em cada linha. Pode-se usar espaços extras entre as *strings* se necessário para obter um comprimento de linha K. Se o número de espaços não for divisível pelo número de intervalos entre as palavras terá de ser adicionado mais espaços entre as palavras à esquerda. A última linha não deve ser justificada, mas alinhada à esquerda.

Considerações: A primeira linha do input conterá os valores de N e K. A segunda linha é a lista com N números inteiros positivos separados por espaço. As próximas linhas de input contém uma palavra do texto, em ordem.
O output será o texto retornado da função.

Linguagem preferencial: Java, Python

Exemplos:

Input:	*Output:*
8 15 *This* *text* *is* *justified.* *Notice* *the* *extra* *spaces.*	*This text is* *justified.* *Notice the* *extra spaces.*

Desafio Nº: 16

Pontos: 10

Título: Algoritmo de ordenação binário estável

Descrição do problema: É dado N nomes (*strings*) de crianças. Por cada criança sabemos o seu nome e a equipa (vermelho ou amarelo). O objetivo é ordenar as crianças consoante os seguinte pontos:

- ✓ Todos os membros da equipa amarela ficam à frente dos da equipe vermelha
- ✓ Por cada duas crianças da mesma equipa mantêm a sua ordem relativa.

O problema terá de ser resolvido com complexidade computacional O(N log N) usando a memória extra O(1).

Considerações: A primeira linha do input é o valor de N. As N linhas seguintes de input é a informação sobre a criança, ou seja, o seu nome e a equipa (V ou A).
O output será o valor obtido da função.

Linguagem preferencial:Java, Python

Exemplos:

Input:	*Output:*
5	
Manuel V	*João A*
Joaquim V	*José A*
João A	*Manuel V*
José A	*Joaquim V*
Miguel V	*Miguel V*

Desafio Nº: 17

Pontos: 10

Título: Interseção entre listas

Descrição do problema: É dado duas listas de inteiros. O objetivo é retornar os elementos em comum. As listas não precisam de ter valores distintos. Se um certo número ocorrer na primeira lista x vezes e na segunda lista y vezes deverá ser retornado esse número min(x,y) vezes. Este problema terá de ser resolvido em O(N1 +N2).

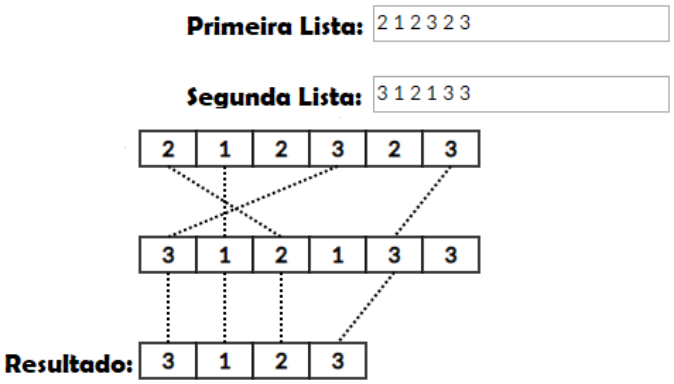
Considerações: A primeira linha do input são os valores de N1 e N2, tamanhos das duas listas a comparar. A segunda linha é a primeira lista com N1 números inteiros separados por espaço. A terceira linha é a segunda lista com N2 números inteiros separados por espaço.
O output será o tamanho da interseção entre as duas listas e os números da interseção.

Linguagem preferencial:Java, Python

Exemplos:

Input1:	*Output1:*
2 2 *1 3* *2 4*	*0*

Input2:	*Output2:*
2 2 *-1 3* *-1 3*	*2* *3 -1*



CAPÍTULO III – DESAFIOS DE NÍVEL DIFÍCIL

Desafio Nº: 18

Pontos: 20

Título: Sudoku Solver

Descrição do problema: O objetivo é criar um programa que resolva o jogo Sudoku.

Considerações: O input consiste em 9 linhas, cada uma contendo 9 dígitos. Uma célula vazia é representada por um 0.
O output são 9 linhas, cada uma contendo 9 dígitos, representando o Sudoku resolvido.
Os inputs têm de garantir exatamente e apenas uma solução.

Linguagem preferencial:Java, Python

Exemplo:

Input:	*Output: 0*
6 0 2 0 8 0 7 4 0	6 1 2 3 8 5 7 4 9
7 0 0 2 9 0 6 0 1	7 3 5 2 9 4 6 8 1
0 0 0 7 0 1 2 0 0	8 4 9 7 6 1 2 3 5
0 0 8 4 0 0 3 0 0	9 6 8 4 1 7 3 5 2
0 0 3 0 2 0 4 0 0	5 7 3 9 2 8 4 1 6
0 0 1 0 0 3 9 0 0	4 2 1 6 5 3 9 7 8
0 0 7 8 0 2 0 0 0	1 9 7 8 3 2 5 6 4
3 0 6 0 4 9 0 0 7	3 8 6 5 4 9 1 2 7
0 5 4 0 7 0 8 0 3	2 5 4 1 7 6 8 9 3

Desafio Nº: 19

Pontos: 20

Título: Número de Kaprekar

Descrição do problema: É dado um número de 4 algoritmos com pelo menos dois algoritmos distintos. Organiza-se os dígitos em ordem decrescente e ordem crescente (adicionando zeros para ajustá-los de forma a ficar um número de 4 dígitos) e subtraia-se o número menor do número maior. De seguida repita o passo anterior. A execução deste processo levará com que se atinja o número fixo: 6174. O objetivo é retornar o número de vezes que o processo é executado até atingir o número 6174.

Considerações: O input consiste num número de 4 dígitos. O output é o valor retornado da função.

Linguagem preferencial:Java, Python

Exemplo:

Input1: 2111	*Output1: 5*
Input2: 9831	*Output2: 7*

Desafio Nº: 20

Pontos: 20

Título: Matriz quadrada

Descrição do problema: É dada uma matriz constituída por zeros e uns. O objetivo é retornar a área da maior submatriz quadrada que contém apenas uns.

Considerações: O input consiste num array com *strings* constituídas por zeros e uns. O output é o valor retornado da função.

Linguagem preferencial:Java, Python

Exemplo:

Input1:["0111", "1111", "1111", "1111"]

Output1:9

Input2:["0111", "1101", "0111"]

Output2:1

Desafio Nº: 21

Pontos: 20

Título: Número Pentagonal

Descrição do problema: É dado um número inteiro positivo N e determina-se quantos pontos existem numa forma pentagonal em torno de um ponto central na N-enésima iteração. O objetivo é retornar o número de pontos que existem no pentágono inteiro na N-enésima iteração.

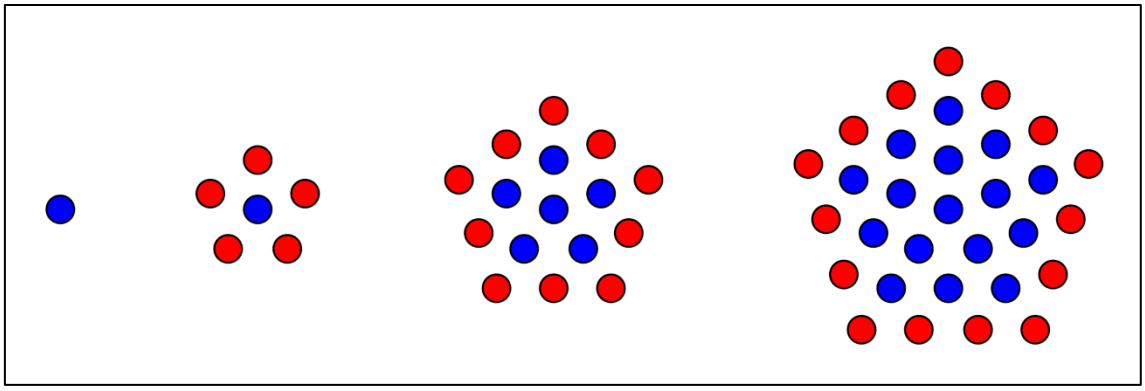


Figura 1: As primeiras quatro iterações

Considerações: O input consiste num número inteiro positivo. O output é o valor retornado da função.

Linguagem preferencial: Java, Python

Exemplo:

Input1:2

Output1:6

Input2: 5

Output2: 51

CAPÍTULO IV – DESAFIOS DE NÍVEL EXTREMO

Desafio Nº: 22

Pontos: 40

Título: Linhas paralelas

Descrição: É dado N pontos distribuídos num plano. O objetivo é calcular o menor número K de linhas paralelas que conseguem cobrir todos os pontos N. O valor do K não pode ser maior que 400.

Consideração: A primeira linha do input contém o número de caso de testes T. Para cada teste, a primeira linha contém um número inteiro positivo N representando o número de pontos. Para cada N linhas seguintes contém as coordenadas de um ponto.
O output para cada caso de teste imprime a resposta numa linha separada.
Tem de se ter em conta também as seguintes condições:

- O valor de T tem de estar compreendido entre 1 e 10.
- O valor de N tem de estar compreendido entre 1 e 30000
- Os pontos têm coordenadas positivas.
- Todos os pontos de cada caso de teste têm de ser distintos.
- A soma de valores de N de todos os casos de testes não pode exceder 30000.
- Para cada caso de teste a resposta não pode ser superior a 400.

Exemplo:

Input:	*Output:*
2	*3*
5	*3*
2 2	
1 2	
2 1	
3 2	
2 3	
7	
1 2	
1 3	
1 5	
2 3	
2 4	
3 4	
4 5	

Desafio Nº: 23

Pontos: 40

Título: Trocar as relações

Descrição: É dado uma árvore com N nós. Cada relação ou é verde ou é amarela. Pode-se escolher qualquer caminho simples na árvore e trocar as cores das relações: de amarelo para verde ou vice-versa. Para algumas das relações sabe-se que o seu estado final pode ser verde ou amarelo. O resto das relações não têm quaisquer restrições. O objetivo é retornar o número mínimo de caminhos a escolher e de todas as soluções obter o menor custo possível.

Consideração: A primeira linha do input contém o número inteiro N. Cada uma das próximas N-1 linhas contêm quatros inteiros *a,b,c, d* que representam uma relação entre nós (a,b) com cor inicial c, onde c é zero se relação é verde ou 1 se for amarelo. O quarto número *d* é o zero se a cor final da relação for verde, 1 se for amarela ou 2 se não houver restrição ao estado final à relação.

- O valor de N tem de estar compreendido entre 2 e 10^5.

Exemplo:

Input1:	*Output1:*
5 *2 1 1 0* *1 3 0 1* *2 4 1 2* *5 2 1 1*	*1 2*

Input2:	*Output2:*
3 *1 3 1 2* *2 1 0 0*	*0 0*

Desafio Nº: 24 - corrigido

Pontos: 40

Título: Telegrafo

Descrição: É dado um texto em que se deseja enviar ao amigo através do telegrafo. Ao usá-lo, pode-se transmitir dois símbolos, . (ponto) e _ (traço raso). Ele e o seu amigo devem encontrar uma maneira de codificar as letras no texto, ou seja, para cada letra associar uma *string* formada por apenas . (ponto) e _ (traço raso). Uma codificação é valida se não puder encontrar duas letras de modo que a *string* associada a uma delas seja um prefixo da *string* associada à outra letra. Por exemplo, se for codificado a letra A por ._, o B pode ser codificado por __._, mas não por ._.
Leva um segundo para enviar um ponto e dois segundos para enviar um traço raso. O objetivo é encontrar uma codificação válida de modo que o tempo total para enviar o texto usando o telégrafo seja o mínimo possível.

Consideração: A primeira linha do input consiste num número inteiro N que representa o número de letras distintas no texto. A segunda linha contém os valores de N que representam as frequências das letras no texto.
O output terá de ser um número inteiro representando o tempo mínimo necessário para enviar o texto sobre o telégrafo usando uma codificação válida.

- O valor de N tem de estar compreendido entre 1 e 750.
- As frequências das letras são inteiros compreendidos entre 1 e 10^5.

Exemplo:

Input	Output
3 2 1 1	9
4 1 2 3 4	27
4 1 2 2 3	22
5 1 1 1 1 1	17
5	34

Input	Output
2 2 2 2 2	

Desafio Nº: 25

Pontos: 40

Título: Rede de Parênteses

Descrição: A sequência de parênteses S é correta se uma das seguintes condições acontecer:

- É vazia
- É uma concatenação de duas sequências corretas
- É da forma (T), em que T é uma sequência correta.

O objetivo é construir uma rede onde cada célula está preenchida com (ou) tal que há exatamente K caminhos a partir do canto superior esquerdo até ao canto inferior direito cujos unimos movimentos possível é ir para baixo e para a direita, e a sequência de parênteses formada pelos quadrados visitados é a sequência correta.

Considerações: O input é o valor de K.
A primeira linha de output deve conter dois inteiros N e M, indicando o número de linhas e o número de colunas da grade, respetivamente. O próximo N deve conter M caracteres cada, onde cada letra deve ser (ou). A rede do output deve conter exatamente K caminhos do canto superior esquerdo até o canto inferior direito, que vai apenas para baixo ou para a direita, de modo que a sequência de parênteses formada a partir do caminho seja uma sequência correta. Se houver várias soluções, qualquer solução válida será aceita.

O valor de K terá de estar compreendido entre 1 e 10^9 e os valores dos outputs N e M têm de estar compreendidos entre 1 e 200.

Exemplo:

Input: *Output:*

4

3 4
()))
((()
()))

Desafio Nº: 26

Pontos: 40

Título: Conto

Descrição: Todos os anos por volta do verão., um rei de estatura baixa, chamado Diogo, dá um banquete no seu castelo no qual todos os cavaleiros de todos os reinos são convidados. Manuel é um dos convidados, e com um verdadeiro cavaleiro, ele cavalga no seu cavalo mágico, chamado Trovosky. Cavalos são proibidos dentro do castelo, portanto o Manuel amarrou o cavalo num local perto fora do castelo. Trovosky é um cavalo calmo desde que tenha algo para comer, mas depois de comer toda a sua comida ao seu redor, fica nervoso e cospe fogo como um dragão. A dado ponto, o Manuel tem de sair do jantar para acalmar o animal de 4 patas.

Para prevenir que o Trovosky cause um fogo, o Manuel precisa de saber a área onde ele pode mover o cavalo. A figura é modelada por:

- A árvore onde o cavalo está amarrado (coordenadas).
- O tamanho da corda L (inteiro positivo).
- A muralha do castelo onde o cavalo não pode saltar (a muralha forma um polígno convexo com N vértices).

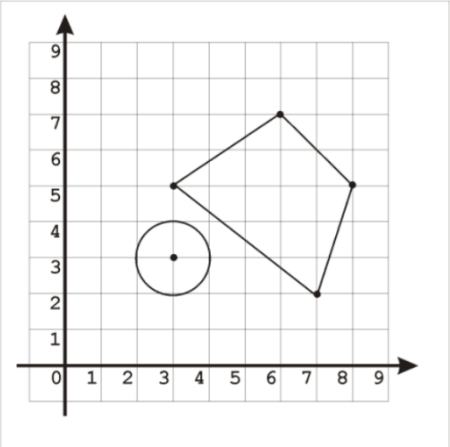
O objetivo é calcular a área da figura onde o Trovosky pode mover-se.

Considerações: A primeira linha de input terá as coordenadas da árvore e o valor de L separados por um espaço. A segunda linha contém o inteiro positivo N. Em cada uma das N linhas seguintes, irá conter as coordenadas dos vértices do polígono, dados no sentido do relógio.

O N tem de estar compreendido entre 3 e 300 e o L entre 1 e 10^5.

Exemplo:

Input:	*Output:*
3 3 1 *4* *3 5* *6 7* *8 5* *7 2*	*3.14159*



CAPÍTULO V – DESAFIOS EXTRA

Desafio Nº: 27

Pontos: 60

Proposto por: Professor Diogo Gomes

Título: Previsão de resultados desportivos

Descrição do problema: As empresas de apostas encontram-se hoje munidas de ferramentas várias de tratamentos de dados estatísticos para o calculo dos prémios. Por outro lado o jogador está apenas munido da sua intuição. Este desafio visa providenciar uma app de auxilio ao jogador.
A app deverá fazer ela própria uso de dados estatísticos e com recurso a algoritmos de IA (exemplo redes neuronais) prever o resultado mais provável de um evento desportivo.

Considerações:
- A equipa deverá reunir ela própria uma DB de resultados históricos (eventualmente através de scrapping de sites)
- Deverá ser definido um ou mais algoritmos
- Deverá ser calculada um previsão do resultado mais provável
- Integração com sites de apostas poderá permitir escolher qual o site que propõe o melhor premio o nosso resultado previsto.

Linguagem preferencial:
- Python

Desafio Nº: 28

Pontos: 60

Proposto por: Professor Diogo Gomes

Titulo: Streaming no ensino da programação

Descrição do problema: Quando se lecciona programação é extremamente util poder partilhar com os alunos o código gerado. Numa geração habituada ao streaming de jogos poderá fazer sentido o streaming de um professor/aluno enquanto programa um exercício.
Ao contrário dos video-jogos a programação não é uma tarefa adequada ao streaming em video, pois é util a quem assiste poder seleccionar o texto transmitido. É pois necessária uma plataforma de streaming de código fonte.

Considerações:
- A plataforma deverá se apoiar em tecnologias Web, sem nunca depender de um editor ele próprio na web, isto é, deverá ser possível fazer stream apartir de um IDE ou até mesmo de um terminal.
- O stream deverá ser distribuído em realtime para todos intervenientes, que podem a qualquer momento seleccionar código do stream e cortar/colar para o seu computador.

Linguagem preferencial:
- Python, nodeJS

Tecnologias preferenciais:
- WebRTC

Desafio Nº: 29

Pontos: 60

Proposto por: Professor Tomás Oliveira Costa

Título: Distribuição equitativa de recursos

Descrição do problema: Desenvolver e implementar um algoritmo que seja capaz de apresentar uma boa solução (não necessariamente a melhor) para o problema descrito a seguir.
Pretende-se colocar N cilindros, de alturas A[1], A[2], ..., A[N], em P pilhas inicialmente vazias, de modo que a maior das alturas das pilhas seja o mais pequena possível.
Cada cilindro tem de ir para uma das pilhas. A altura de cada pilha é a soma das alturas dos cilindros que lá foram colocados. No programa considere que N <= 200 e que P <= 30.

Considerações: A altura mínima da pilha mais alta não pode ser inferior a ceil(S/P), onde S é a soma das alturas de todos os cilindros. A qualidade da solução encontrada será dada pela diferença entre a maior das alturas das pilhas e o valor ceil(S/P): quanto mais próximo de zero melhor.
Em caso de empate, o tempo de execução do programa servirá para desempatar. Por exemplo, para N=33, e sendo A[1]=2, A[2]=3, A[3]=5, ... (A[k] é o k-ésimo número primo), e para P=7, temos S=2+3+...+137=1988, pelo que S/P=284.
Neste caso pode ser que exista uma solução em que todas as pilhas têm a mesma altura.
E neste caso assim é:

* pilha 1 (284): 2 5 17 23 47 59 131
* pilha 2 (284): 3 43 101 137
* pilha 3 (284): 7 73 97 107
* pilha 4 (284): 11 29 37 41 53 113
* pilha 5 (284): 13 61 83 127
* pilha 6 (284): 19 67 89 109
* pilha 7 (284): 31 71 79 103

O problema deve ser resolvido para o caso geral, mas para esta competição e de modo a que seja fácil gerar exemplos completos sem grande esforço, considere que A[k] é, tal como no exemplo acima, o k-ésimo número primo.
Deste modo o programa deverá ser invocado apenas com dois argumentos: o valor de N e o de P.
Nota final: o problema (bin packing) é computacionalmente difícil, pelo que se devem evitar soluções que recorram a força bruta.

Desafio Nº: 30

Pontos: 90

Proposto por: Dellent

Título: Optimização do processo de download de recursos de servidor a partir de equipamentos de IoT dentro de uma mesma rede.

Descrição do problema: Considere-se um cenário com 2 ou mais equipamentos IoT instalados dentro de uma mesma rede ethernet e que se ligam a um mesmo servidor para trocar dados e fazer downloads de recursos.
Os equipamentos descarregam recorrentemente ficheiros que poderão ter algumas dezenas de MB. Numa rede com 10 equipamentos ligado à mesma rede pode acontecer que em determinado momento todos os equipamentos necessitem de descarregar o mesmo ficheiro do servidor. Nesse momento será desnecessariamente imposta uma grande carga tanto na rede local, como no servidor.
Pretende-se melhorar o processo de download de recursos pesquisando primeiro se o recurso pretendido está num dos equipamentos na mesma rede e só em caso de não ser encontrado em algum equipamento da rede local, recorrer ao download a partir do servidor.

Considerações:
- A gestão de rede não está sob nosso controlo;
- A configuração de IP é feita de forma automática por DHCP;
- Os equipamentos não possuem interfaces gráficos ou de interacção;
- Todos os equipamentos ligam-se a um servidor que poderão usar para persistir dados e eventualmente, trocar informações entre eles através desse ponto comum;
- Todos os downloads realizados pelos equipamentos são verificados por comparação de MD5 após o download. Só então é que o download é validado;
- Pode haver alturas em que o pedido de download de um mesmo recurso do servidor pode ocorrer aproximadamente na mesma altura em vários equipamentos;
- Ter em consideração que, quando um recurso é solicitado pode já estar a ser descarregado por um dos equipamentos;
- Ter em consideração que o download pode falhar em qualquer momento, tanto do servidor como entre equipamentos. O download pode falhar por, pelo menos, uma das seguintes razões:
  - O equipamento que contém o recurso foi desligado;
  - O equipamento que está a descarregar o conteúdo foi desligado;
  - O servidor foi desligado;

  Os equipamentos deverão ter a capacidade de resumir ou reiniciar o download a partir da fonte mais próxima.

Linguagem preferencial:
- Python