

# Teoria da Informação

**Licenciatura em Engenharia Informática**

**2023**

**Relatório | Trabalho Prático – 2**

**Descompactação de Ficheiros ‘gzip’**

**Trabalho realizado por:**

**Eduardo Marques - 2022231584**

**João Cardoso - 2022222301**

**Sérgio Marques - 2022222096**

# Índice

Exercício 1.....	2
Exercício 2.....	2
Exercício 3.....	2
Exercício 4.....	3
Exercício 5.....	3
Exercício 6.....	3
Exercício 7.....	4
Exercício 8.....	4

## Exercício 1

Neste primeiro exercício é lido o formato do bloco, calculando os valores de “HLIT”, “HDIST” e “HCLEN”, de acordo com a estrutura do bloco. Para isso, para cada uma destas variáveis, lemos um número de bits indicado e adicionamos um valor previamente definido ao resultado desta operação.

## Exercício 2

Para a realização deste exercício, é criado um array onde vão ser adicionados os comprimentos dos códigos do “alfabeto de comprimentos de códigos”. Estes comprimentos são calculados lendo sequências de 3 bits de acordo com a ordem fornecida no enunciado.

## Exercício 3

Este exercício tinha como objetivo converter os comprimentos dos códigos do alfabeto de comprimentos de códigos em códigos de Huffman. Para isso, estes comprimentos foram convertidos, inicialmente, em valores que foram, posteriormente, convertidos em números binários.

De modo a converter os comprimentos, foi contado o número de vezes que cada comprimento ocorre no alfabeto. De seguida, somamos cada contador (relativo a um determinado comprimento) ao valor obtido anteriormente e multiplicamos esta soma por 2, andando um bit para a esquerda. Por último, armazenamos cada valor ao índice correto do array de valores convertidos.

Com o array de valores convertidos já construído, convertemos cada um destes valores para binário e é verificada a necessidade de acrescentar, ou não, 0’s no início do número binário, para que o seu comprimento corresponda ao comprimento armazenado no array. Assim, serão construídos corretamente os código de Huffman pretendidos.

Por último, é ainda construída, a partir destes códigos, a árvore de Huffman deste alfabeto.

## Exercício 4

O objetivo deste exercício era calcular os comprimentos dos códigos referentes ao alfabeto de literais/comprimentos. Para isso, foi percorrida a árvore de Huffman de “HCLN”, construída no exercício anterior. De modo a serem encontradas as folhas desta árvore, foi utilizada a função “nextNode” do ficheiro “huffmantree.py”, que tem em conta o nó atual da árvore e o próximo bit, que indicará a direção de pesquisa.

Se o valor da posição da folha for diferente de 16, 17 e 18, este valor será armazenado no array de comprimentos. Caso contrário, terão que ser lidos bits extra, de modo a calcular o número de vezes que uma operação ocorre.

Para o valor 16, esta operação equivale à colocação, no array de comprimentos, do valor que se encontra na posição anterior do array. No caso dos valores 17 e 18, será colocado no array de comprimentos o valor 0.

## Exercício 5

Para a realização deste exercício, foi adotado o método utilizado no exercício anterior, usando a função “comprimentosCodigos”, para calcular os comprimentos dos códigos do alfabeto de distâncias.

## Exercício 6

Neste exercício, era-nos pedido que fossem determinados os códigos de Huffman referentes aos alfabetos de literais/comprimentos e de distâncias. Para isso, foram utilizadas novamente as funções usadas no exercício 3. Por fim, foram, também, criadas as árvores de Huffman destes dois alfabetos.

## Exercício 7

Este exercício tem como objetivo construir o array de valores ASCII dos caracteres que serão escritos no ficheiro de texto. Para isso, foram criados dois dicionários que contêm, respetivamente, informações sobre os alfabetos de literais/comprimentos e de distâncias (símbolo, comprimento mínimo e número de bits extra a ler).

De seguida, foi percorrida a árvore de Huffman até ser encontrada a posição 256 (que indica o fim do bloco). Posição essa que é determinada através da função “nextNode” do ficheiro “huffmantree.py”.

Se o valor da posição da folha encontrada for menor ou igual a 255, estaremos a falar de um literal e, o seu valor será armazenado no array de valores.

Caso contrário, serão lidos alguns bits extra, que vão ser úteis para o cálculo do comprimento e da distância a recuar no array. Depois de serem calculados estes parâmetros, será calculada a posição do array de valores até onde se deve recuar para encontrar o primeiro valor a colocar no array e, será adicionado ao array um certo número de valores, que corresponderá ao valor do comprimento calculado.

## Exercício 8

Neste último exercício, é criada uma função que irá receber o array que contém os valores em ASCII e irá convertê-los no carácter correspondente. Ao juntar estes caracteres, será criada a mensagem que vai ser, posteriormente, escrita no ficheiro de texto que terá o nome do ficheiro em análise.