

Base64

Programação de Computadores I

Ciência da Computação

Prof. Daniel Saad Nogueira Nunes



**INSTITUTO
FEDERAL**
Brasília

1 Introdução

O método de codificação e decodificação **Base64** permite a transferência de qualquer conteúdo binário através de um texto em ASCII que contém apenas caracteres imprimíveis. Ele é frequentemente empregado quando se quer transferir *streams* sobre um alfabeto binário através de um texto, como páginas HTML.

2 Objetivos

O objetivo deste trabalho é a implementação de um programa capaz de:

- Codificar um arquivo binário qualquer em um arquivo no formato Base64.
- Decodificar um arquivo Base64 para obtenção do arquivo binário original.

2.1 Especificação

De acordo com a RFC2045 [FB96], que estabelece a especificação das extensões de correio eletrônico multipropósito (MIME), o padrão Base64 permite a representação de octetos (conjuntos de bytes) através de um alfabeto que consiste de apenas 65 símbolos, sendo um deles utilizado apenas para fins de preenchimento (padding).

O processo de codificação é disposto da seguinte maneira: grupos de 3 octetos são concatenados formando uma sequência de 24 bits, a qual pode ser encarada como 4 grupos de 6 bits. Cada um destes grupos é substituído pelo caractere correspondente da Tabela 1.

Tabela 1: Tabela de codificação/decodificação Base64.

Valor	Codificação	Valor	Codificação	Valor	Codificação	Valor	Codificação
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v		
14	O	31	f	48	w	(pad)	=
15	P	32	g	49	x		
16	Q	33	h	50	y		

Por exemplo, suponha que os primeiros três bytes de um arquivo sejam:

01011100 11011010 10111100

Estes bytes são encarados como uma sequência de 24 bits que é dividida nos seguintes grupos de 4 bits:

$$\underbrace{010111}_X \underbrace{001101}_N \underbrace{101010}_q \underbrace{111100}_8$$

Substituindo o valor de cada sequência de 6 bits pela codificação da Tabela 1 obtém-se a sequência em Base64 **XNq8**.

O processo de decodificação é simétrico, isto é, a cada 4 símbolos Base64, são produzidos 3 octetos.

A única ressalva é quando o arquivo a ser codificado possui um tamanho que não é múltiplo de 3, fazendo com que não seja possível formar um grupo de três bytes no fim do arquivo. Neste caso, o caractere de preenchimento '=' deve ser utilizado. Quando o tamanho do arquivo deixa resto 2 na divisão por 3, são utilizados dois caracteres de preenchimento. Já quando o tamanho do arquivo deixa resto 1 na divisão por 3, apenas um caractere de preenchimento precisa ser utilizado. Utilizando os textos *{cavalo, avalo, valo, alo, lo, o}*, a Tabela 2 fornece as seguintes codificações e seus respectivos preenchimentos.

Tabela 2: Textos e respectivos preenchimentos.

Texto	Codificação
cavalo	Y2F2YWxv
avalo	YXZhbG8=
valo	dmFsbw==
alo	YWxv
al	YWw=
o	bw==

2.2 Divisão em Módulos

Para realizar esta tarefa, além do arquivo que provê a função **main**, deverão ser criados os arquivos **encode.c**, **encode.h**, **decode.c** e **decode.h**, que representam a implementação e definição das funções de codificação e decodificação do método Base64.

Juntamente com os códigos, deverá ser produzida uma documentação instruindo o usuário a compilar o código e gerar o executável final.

2.3 Entrada

O seu programa deve receber, via argumentos de linha de comando, três parâmetros além do nome do executável:

- Modo de operação.
- Arquivo de entrada.
- Arquivo de saída.

O modo de operação pode assumir os valores “-c” ou “-d”. O primeiro indica que deverá ser feita a codificação de um arquivo no formato Base64, já o segundo informa que deve ser realizada a tradução de um arquivo Base64 no arquivo original. O arquivo de entrada corresponde ao caminho do arquivo a ser codificado/decodificado. O arquivo de saída, representa o caminho do arquivo a ser salvo de acordo com o modo de codificação e o arquivo de entrada.

Abaixo seguem alguns exemplos de entrada e o resultado esperado:

- `./base64 -c flor.jpg flor-base64.txt`: codifica a imagem `flor.jpg` no arquivo texto `flor-base64.txt`.
- `./base64 -d texto-em-base-64.txt dog.gif`: decodifica o arquivo em Base64 `texto-em-base-64.txt` e salva o conteúdo no arquivo `dog.gif`.

2.4 Saída

Deve ser produzido um arquivo de saída contendo a codificação/decodificação do método Base64.

3 Critérios de correção

Deve ser utilizada a linguagem de programação C para a implementação do caça-palavras. A Tabela 3 expõe os critérios de avaliação.

Tabela 3: Tabela de pontuação	
Critério	Nota
Correção da Implementação	70
Documentação	10
Modularização	20
Total	100

Para validação da correção do algoritmo, testes automatizados serão realizados, então é **crucial** que a saída esteja conforme o especificado.

Serão descontados pontos dos códigos que não possuírem identificação.

4 Considerações

- Este trabalho deve ser feito **individualmente**.
- O trabalho que não compilar não será avaliado.
- Cópias e plágio serão avaliados automaticamente com nota 0 para os envolvidos. Medidas disciplinares também serão tomadas.
- O trabalho deve ser entregue dentro de uma pasta zipada com a devida identificação do aluno através da plataforma edmodo.

Referências

- [FB96] Ned Freed and Nathaniel Borenstein, *Multipurpose internet mail extensions (mime) part one: Format of internet message bodies*, Tech. report, 1996.