



OBJETIVO: Exercitar a manipulação de arquivos/dados binários através da leitura de um arquivo PNG (*Portable Network Graphics*).

QUESTÃO ÚNICA

Assim como todos os padrões de arquivos binários, o arquivo PNG segue um padrão bem estabelecido que, se seguido, permite acessar suas informações. Mais especificamente, um arquivo PNG possui o seguinte formato:

| Assinatura PNG | Chunk 1 (IHDR) | Chunk 2 | Chunk 3 | ... | Chunk <i>n-1</i> | Chunk <i>n</i> (IEND) |
|----------------|----------------|---------|---------|-----|------------------|-----------------------|
|----------------|----------------|---------|---------|-----|------------------|-----------------------|

Onde a “Assinatura PNG” é igual em todos os arquivos PNG e possui a string (8 bytes):

`"\x89\x50\x4E\x47\x0D\x0A\x1A\x0A"`.

Curiosidade:

Na assinatura acima, os caracteres `\x50 \x4E \x47` são, respectivamente, “PNG”. Portanto, se você abrir um arquivo PNG usando um editor de texto comum, a string “PNG” sempre aparecerá após o primeiro caractere.

Após a assinatura, um arquivo PNG é composto de vários “Chunks” (do inglês, “Pedacos”). Cada Chunk possui o formato especificado a seguir (fonte: PNG Specification – <http://www.w3.org/TR/PNG/>).

| Length (4 bytes) | Chunk Type (4 bytes) | Chunk Data (“Length” bytes) | CRC (4 bytes) |
|------------------|----------------------|-----------------------------|---------------|
|------------------|----------------------|-----------------------------|---------------|

Onde “Length” é o tamanho do chunk, “Chunk Type” é o tipo deste chunk, “Chunk Data” possui os dados do chunk e, por último, o “CRC” é usado para verificar se os dados do chunk estão corretos.

A especificação do PNG prevê diversos tipos de chunks (incluindo um chunk para os dados da imagem propriamente dita). A tabela a seguir, lista alguns dos principais:

Principais Chunks do PNG (existem outros)

Note que os tipos dos chunks (Chunk Type) são identificados por uma string de 4 caracteres. As que começam com letra maiúscula são obrigatórias (presentes em todos os PNGs).

| Chunk Type | Descrição |
|------------|--|
| IHDR | Contém a largura, altura e outras informações da imagem. Obrigatoriamente este deve ser o primeiro chunk do PNG. |
| PLTE | Contém a paleta de cores. |
| IDAT | Contém os dados da imagem! Provavelmente será o chunk de maior tamanho. |
| IEEND | Marca o final do arquivo PNG (tem tamanho zero!). Tem que ser o último. |
| bKGD | Cor padrão do fundo da figura. |
| tEXt | Armazena texto no formato de um par <nome>=<valor>. |
| tIME | Indica a data de última modificação do arquivo. |

Curiosidade:

Se você abrir qualquer PNG em um editor de texto comum, ele terá pelo menos as palavras IHDR, PLTE, IDAT e IEND.

Para cada um dos tipos (Chunk Type), a especificação define o formato dos dados presentes no Chunk Data. Por exemplo, se o chunk for do tipo “IHDR”, o Chunk Data terá o seguinte formato (fonte: PNG Specification – <http://www.w3.org/TR/PNG/>):

Chunk Data para o Chunk do tipo IHDR

Para pegar as informações dos chunks e dos dados deles, basta criar uma estrutura e usar o `fread` para pegar os dados do arquivo de acordo com a estrutura criada (da mesma forma como feito no exercício anterior).

| Campo | Tamanho |
|--------------------|---------|
| Width | 4 bytes |
| Height | 4 bytes |
| Bit depth | 1 byte |
| Colour type | 1 byte |
| Compression method | 1 byte |
| Filter method | 1 byte |
| Interlace method | 1 byte |

Mas atenção! Aviso muito importante! Não deixe de ler! Todos os números inteiros de um arquivo PNG (e.g., tamanho do chunk, largura e altura do IHDR, etc) estão em formato “network byte order” (*big-endian*), o que significa que eles provavelmente precisarão ser “convertidos” para a sua arquitetura atual. Para fazer isso, usa-se (sempre) a função `ntohl`, que converte um inteiro da “rede” para o formato “local”. Por exemplo, para converter o tamanho de um chunk:

```
fread(png_chunk, sizeof(struct png_chunk_hdr), 1, png_file); // Lê um chunk
printf("    --> Tamanho: %d\n", ntohl(png_chunk->length)); // Converte e imprime o tam.
```

No linux, o `ntohl` está disponível automaticamente ou fazendo um “include” do `arpa/inet.h`, mas no windows, talvez seja necessário fazer um “include” do `winsock2.h`.

Fim do aviso muito importante. O restante a seguir também é muito importante ...

Sua missão, é abrir um arquivo PNG, cujo nome será passado pela linha de comando (argv[1]), e imprimir o tamanho e o tipo de todos os chunks presentes nele, na ordem em que aparecem. Se for um chunk do tipo "IHDR", imprima a largura e a altura da imagem PNG.

Exemplo de Execução:

```
./a.out Teste.png
Lendo chunk 1:
--> Tamanho: 13
--> Tipo:    IHDR
--> Largura: 160
--> Altura:  80
Lendo chunk 2:
--> Tamanho: 6
--> Tipo:    bKGD
Lendo chunk 3:
--> Tamanho: 9
--> Tipo:    pHYS
Lendo chunk 4:
--> Tamanho: 7
--> Tipo:    tIME
Lendo chunk 5:
--> Tamanho: 25
--> Tipo:    tEXt
Lendo chunk 6:
--> Tamanho: 785
--> Tipo:    IDAT
Lendo chunk 7:
--> Tamanho: 0
--> Tipo:    IEND
```

Dicas (algoritmo):

- 1 Crie a estrutura do chunk incluindo apenas o "tamanho" e o "tipo", pois os dados do chunk são variáveis.
- 2 Crie a estrutura do chunk IHDR de acordo com a tabela passada anteriormente
- 3 Aloque memória para as estruturas criadas (malloc);
- 4 Abra o arquivo PNG;
- 5 Pule 8 bytes (assinatura do PNG – fseek);
- 6 Enquanto não for o fim do arquivo PNG (feof)
 - 6.1 Leia do arquivo a estrutura do chunk e salve na memória alocada (fread)
 - 6.2 Imprima o tamanho e o tipo do chunk
 - Dica 1: não esqueça do `ntohl` ao imprimir o tamanho
 - Dica 2: para imprimir apenas 4 caracteres de um vetor de caracteres, use o modificador `% .4s`
 - 6.3 Se for um chunk do tipo "IHDR"
 - 6.3.1 Leia do arquivo a estrutura do chunk IHDR (fread)
 - 6.3.2 Imprima a largura e a altura da figura (não esqueça de usar o `ntohl`)
 - 6.3.3 Pule 4 bytes do arquivo (para pular o CRC – fseek)
 - 6.4 Se não, se for um chunk do tipo "IEND"
 - 6.4.1 Quebre o loop
 - 6.5 Se não for nenhum dos dois tipos acima
 - 6.5.1 Pule `ntohl(png_chunk->length)+4` bytes (fseek)
- 7 Libere as memórias alocadas
- 8 Feche o arquivo PNG

ENTREGA DO LABORATÓRIO

Este laboratório é remoto e deve ser entregue até 20/06. Para entregar, envie o código-fonte para horacio@icomp.ufam.edu.br com o assunto "Entrega do 7o Laboratório de LPA".