

## Projeto: Compressão de Imagens

**Data de entrega:** Projeto 1: 14/10/2025

**Grupos de 2 alunos.**

### Objetivos:

O objetivo principal do projeto é colocar em prática conceitos teóricos abordados em sala de aula relacionados a técnicas de codificação de mídias. Para isso, adotaremos a estratégia **PBL** (*Problem Based Learning*) visando implementar algumas técnicas de compressão e descompressão – a saber. O desenvolvimento do projeto será discutido aula a aula. Aqui descreve-se os procedimentos de modo geral.

**Variações e especificidades podem ser definidos durante as aulas.**

Cada grupo deverá desenvolver um compressor para imagens e seu respectivo descompressor. O **compressor** deve aceitar como entrada uma imagem (segundo modelo abaixo). Como saída o compressor deve: 1) gerar um arquivo binário que represente a imagem de modo comprimido; 2) apresentar a taxa de compressão atingida.

O **descompressor** deve receber como entrada um arquivo binário armazenado em disco, o qual deve ter sido produzido pelo correspondente compressor. Como saída o descompressor produz uma imagem com qualidade similar à imagem original.

As técnicas de compressão a serem desenvolvidas são, no mínimo: 1) compressão sem perdas por método entrópico (codificação por diferenças + método estatístico); 2) compressão com perdas com o apoio de: transformação do espaço de cor e *subsampling*.

### Importante

- 1) Projetos envolvidos em casos de plágio e/ou cópia receberão nota zero – todos os envolvidos.
- 2) Só serão avaliados os projetos que:
  - a. apresentarem implementações funcionando para compressor e para descompressor;
  - b. apresentarem compressor e descompressor devidamente implementados como programas separados;
  - c. apresentarem códigos que compilem sem erros.

### Critérios de avaliação:

O projeto será avaliado segundo os critérios abaixo:

- Taxa de compressão: 30%
- Qualidade da imagem recuperada: 30%
- Corretude, adequação e aderência: 20%
- Facilidade de uso (menus e dicas de uso): 10%
- Documentação (interna e externa) adequada e robusta: 10%

### Considerações sobre o projeto:

- Cada grupo deverá desenvolver um projeto original. Cópias, mesmo parciais, não serão toleradas!!
- A implementação deverá ser feita em C, padrão C.99, compilador C GNU. Terminal Linux ou Mac já possuem GCC nativamente. Para Windows pode-se utilizar/installar o Cygwin.
- O programa deverá operar por linha de comando (sem interface gráfica). Sem uso de bibliotecas externas ao padrão.

- Algumas bibliotecas próprias do projeto poderão ser fornecidas pelo professor.
- A implementação das técnicas deverá primar pela busca da maior taxa de compressão com melhor qualidade possível.
- Deverão ser entregues: os arquivos contendo o código-fonte (arquivos .c e .h), makefile para compilação e arquivo *readme*.
- A documentação interna deve conter:
  - os componentes do grupo (nome completo e número USP),
  - documentação das funções (entrada, saída e objetivo), comentários pertinentes das decisões relevantes não triviais.
  - O arquivo *readme* deverá conter: nomes dos componentes do grupo, explicações sobre modo de uso, justificativas das estruturas de dados e algoritmos utilizados na implementação.
- **Modo de entrega:** um dos componentes do grupo deverá fazer *upload* no Tidia de arquivo .zip contendo: Makefile, códigos-fonte (arquivos .c e .h), e arquivo *readme*.

### Modelo (simplificado) de imagem

- Arquivos BMP, sem compressão.
- 24 bits para cor.
- Altura e largura sempre múltiplos de 8.
- Dimensões mínimas: 8 x 8 pixels.
- Dimensões máximas: 1280 x 1024 pixels.

### Estruturas para Bitmaps:

```
typedef struct                /**** BMP file header structure ****/
{
    unsigned short bfType;    /* Magic number for file */
    unsigned int   bfSize;    /* Size of file */
    unsigned short bfReserved1; /* Reserved */
    unsigned short bfReserved2; /* ... */
    unsigned int   bfOffBits; /* Offset to bitmap data */
} BMPFILEHEADER;
```

/\* bfType deve ser = "MB" \*/

```
typedef struct                /**** BMP file info structure ****/
{
    unsigned int   biSize;    /* Size of info header */
    int            biWidth;   /* Width of image */
    int            biHeight;  /* Height of image */
    unsigned short biPlanes;  /* Number of color planes */
    unsigned short biBitCount; /* Number of bits per pixel */
    unsigned int   biCompression; /* Type of compression to use */
    unsigned int   biSizeImage; /* Size of image data */
    int            biXPelsPerMeter; /* X pixels per meter */
    int            biYPelsPerMeter; /* Y pixels per meter */
    unsigned int   biClrUsed; /* Number of colors used */
    unsigned int   biClrImportant; /* Number of important colors */
} BMPINFOHEADER;
```

### Leitura de Bitmaps (exemplo FileHeader):

```
void leituraHeader(FILE *F, BITMAPFILEHEADER *H){
/*F é o arquivo Bitmap que deve ter sido “lido” do disco*/
```

```
fread(&H->Type,sizeof (unsigned short int),1,F);  
fread(&H->Size,sizeof (unsigned int),1,F);  
fread(&H->Reserved1,sizeof (unsigned short int),1,F);  
fread(&H->Reserved2,sizeof (unsigned short int),1,F);  
fread(&H->OffBits,sizeof (unsigned int),1,F);  
}
```

**Bom trabalho!**