RESUMO: Estas são umas das principais funções que usamos no projeto, tem mais algumas mas são de instalação de mouse, teclado, vídeo bem como a desinstalçao dos mesmos para o alegro reconhecer! Ver as funções Ainicia() e Afinaliza() da classe CXadrez! Basicamente como um jogo escrito em alegro 4 funciona: IMPRIME BITMAP NO BUFFER, BUFFER IMPRIME NA TELA, CARREGA ALGUM MIDI OU SAMPLE. Por exemplo: Para mudar da tela inicial para a tela do xadrez a logica seria: VERIFICA SE VC CLICOU NO BOTAO DIREITO DO MOUSE E O PONTEIRO DO MOUSE ESTÁ EM CIMA DA AREA DO BOTAO! CASO SIM, DESTRUA ESSE BITMAP (ESSA IMAGEM) E CARREGA A IMAGEM TAL (NESSE CASO A DO TABULEIRO, PEÇAS E O SCORE QUE CRIEI). E o TEXTO? Mais simples ainda, utiliza a função textprintf e imprime qualquer texto na posição x e y como quiser! O alegro trata a tela como uma matriz de pixel, então o x e y de um bitmap ou de um texto é em relação as pixels da tela, essa é outra razão para usarmos a técnica do buffer que citei mais acima. Não precisa explicar tudo na apresentação, é pouco tempo, se atentem aos slides e relatório, erros de português, ele olha muito isso. Boa sorte, pra vocês! :)

Algumas explicações:

O projeto utiliza a biblioteca Allegro 4, diferentemente da biblioteca Allegro 5 aqui temos que trabalhar sempre com sprites e com impressões na tela.

O Allegro trata todos os gráficos que podem ser exibidos na tela ou carregados na memória como bitmaps. Bitmaps são matrizes de pixels, em que cada valor indica uma cor. Para declarar um bitmap, digite:

BITMAP \*nome\_do\_bitmap;

Utilizamos isso constantemente, toda imagem que vc vê é um bitmap na tela.

Imprimir na tela diretamente muitas vezes da bug, por isso sempre criamos um bitmap chamado BUFFER (ver arquivo CTabuleiro.cpp linha 157) e vamos imprimindo apenas nele, e ele na tela!!

Load\_bitmap(caminho do arquivo, paleta de cores);

A função acima serve pra carregar um bitmap, olhe na linha 160 do arquivo CTabuleiro.cpp (todo Sprite da gente é um bitmap, os sprites são as imagens, os bitmaps dos jogos)

void blit(BITMAP \*source, BITMAP \*dest, int source\_x, int source\_y, int dest\_x, int dest\_y, int width, int height);  
  
Copia uma área retangular, de largura **width** e altura **height**, da coordenada (**source\_x**, **source\_y**) do bitmap apontado por **source** para a coordenada (**dest\_x**, **dest\_y**) do bitmap apontado por **dest**.

void draw\_sprite(BITMAP \*bmp, BITMAP \*sprite, int x, int y);  
  
Copia inteiramente o bitmap apontado por **sprite** na coordenada (**x**, **y**) do bitmap apontado por **bmp**. Equivalente a blit(**sprite**, **bmp**, 0, 0, x, y, **sprite->w**, **sprite->h**).

 Caso o valor de key[KEY\_TECLA] seja verdadeiro, a tecla está sendo pressionada. Por exemplo:

if (key[KEY\_D])

printf("Voce pressionou a tecla 'D'\n");

mouse\_x();

mouse\_y();

As funções acima retornam o valor inteiro onde está o ponteiro do mouse em relação ao buffer (valor x ou y como expliquei no resumo).

O bit 0 de **mouse\_b** guarda o estado do botão esquerdo, o bit 1 o do botão esquerdo, e o bit 2 o do botão do meio.

if (mouse\_b & 1)

printf("Botão esquerdo do mouse pressionado\n");

if (!(mouse\_b & 2))

printf("Botão direito do mouse não pressionado\n");

if (mouse\_b & 4)

printf("Botão do meio do mouse pressionado\n");

SAMPLE É DIFERENTE DE MIDI! O SAMPLE É UMA MUSICA DE EFEITO, POR EXEMPLO QUANDO A GENTE APERTA NO BOTÃO DE INICIAR O JOGO, AQUELE BARULHO É UM SAMPLE, JÁ AS MUSICAS QUE SÃO TOCADAS SÃO MIDIS.

**MIDI \*load\_midi(char \*filename);**  
  
Carrega um arquivo MIDI, retornando um ponteiro para uma estrutura **MIDI**. Caso ocorra um erro, o valor **NULL** será retornado.

**int play\_midi(MIDI \*midi, int loop);**  
  
Toca a MIDI especificada por **midi**, parando de tocar qualquer música que estivesse sendo tocada anteriormente. Se a flag **loop** estiver setada, a música será tocada até que a função seja novamente chamada para tocar outra música, ou a função [stop\_midi](http://equipe.nce.ufrj.br/adriano/c/apostila/allegro/docs/funcoes.html" \l "sound_stop_midi) seja chamada. Caso a flag **loop** não esteja setada, a música irá parar de tocar ao alcançar o final do MIDI. Retorna um valor diferente de zero se um erro ocorrer.

**SAMPLE \*load\_sample(char \*filename);**  
  
Carrega um arquivo sample, retornando um ponteiro para uma estrutura **SAMPLE**. Caso ocorra um erro, o valor **NULL** será retornado. Aceita arquivos do tipo WAV (mono e stereo) e VOC (mono) em formato 8 e 16 bits.

**int play\_sample(SAMPLE \*spl, int vol, int pam, int freq, int loop);**  
  
Inicializa o sample especificado por **spl** utilizando determinados argumentos. O argumento **vol** determina o volume (valores de 0 até 255), o argumento **pan** determina o balanço (valores de 0 até 255), o argumento **freq** determina a frequência com que o sample é tocado (o valor 1000 faz com que o sample seja tocado na mesma frequência que foi gravado; o valor 2000 é o dobro da frequência, etc.) e o argumento **loop** informa se o Allegro deve repetir o sample infinitamente ou não.

Assim como para os bitmaps e paletas, o Allegro define um tipo FONT, que contém a descrição de fontes que podem ser desenhadas na tela. Para declarar uma fonte, digite:

FONT \*nome\_da\_fonte;

void textprintf(BITMAP \*bmp, FONT \*f, int x, int y, int color, char \*fmt, ...);  
  
Escreve um texto formatado (no mesmo estilo da função printf) no bitmap apontado por bmp, na coordenada (x, y), utilizando a fonte apontada por f e a cor color.