UNIVERSIDADE DE BRASÍLIA
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

116394 ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES

Trabalho Programação Assembler

OBJETIVO

Este trabalho objetiva a prática da programação em *assembler* do RiscV. O trabalho consiste em implementar o algoritmo de Bresenham para desenho de linhas no ambiente gráfico do RARS.

DESCRIÇÃO

A. Display gráfico mapeado em memória

- O RARS oferece, dentre as suas ferramentas de apoio ao desenvolvimento de aplicações em *assembler* RiscV, uma janela gráfica baseada em *pixels*. Suas principais características são as seguintes:
  - Resolução configurável, *default* 512 x 256 *pixels*.
  - Cada *pixel* é representado por uma palavra de 32 bits, no formato RGB, um *byte* para cada cor. *Red* = 0x00FF0000, *Green* = 0x0000FF00, *Blue* = 0x000000FF.
  - *Display* mapeado em memória. O ponto superior esquerdo da tela corresponde ao pixel com coordenadas (0, 0). A coordenada *x* cresce para a direita e a coordenada *y* cresce para baixo.
  - O endereço correspondente ao primeiro pixel é configurável. Opções: *global data, global pointer, static data, heap, memory map.*
  - O desenho de um pixel na tela é realizado pela escrita de uma palavra contendo a descrição de sua cor RGB na posição de memória correspondente.

B. Função ponto(x, y)

Função que desenha um ponto na tela gráfica a partir de suas coordenadas x e y. A cor do ponto deve ser definida na área de dados. A função deve calcular o endereço de memória correspondente ao ponto *(x, y)*. Deve-se armazenar na área de dados a largura e a altura da imagem em *pixels*, assim como o endereço inicial da imagem na memória. Os parâmetros são como segue:

```
.data
color:      .word 0xFFFF        # azul turquesa
dx:         .word 64            # linha com 64 pixels
dy:         .word 64            # 64 linhas
org:        .word 0x3000        # endereço da origem da imagem (heap)
```

O endereço é calculado pela expressão: *end = 4*x + 4*(y*dx) + org*
Sugere-se criar uma função *getaddress(x, y)* que, dado os valores de *x* e *y* retorna o endereço em memória correspondente ao *pixel*.

C. Algoritmo de Bresenham

```
line(x0,y0, x1,y1) {

  dx=x1-x0;
  dy=y1-y0;

  D = 2*dy - dx;
  ponto(x0,y0);
  y=y0;

  for x from x0+1 to x1
    if (D > 0) {
      y = y+1;
      ponto(x,y);
      D = D + (2*dy-2*dx);
    }
    else {
      ponto(x,y);
      D = D + (2*dy);
    }
}
```

D. Interface com o usuário

- Utilizando as chamadas do sistema, escrever uma mensagem solicitando que o usuário entre com os valores (x0, y0) e (x1, y1) da linha a ser desenhada.

E. Configuração do bitmap display do RARS
- Configuração de memória compacta
- Unit Width in Pixels: 4
- Unit Height in Pixels: 4
- Display Width in Pixels: 256
- Display Height in Pixels: 256
- Base address for display: 0x3000 (heap)

ENTREGA

Entregar no Moodle em um arquivo compactado:

- Relatório de implementação:
  - *cabeçalho*: com título do trabalho, nome e matrícula do aluno, identificação da turma
  - *objetivo*: sumarize os objetivos principais do trabalho
  - *documentação do código*: indique quais as funções implementadas (todas), seus parâmetros e funcionamento.
- código assembler: arquivo asm