Projeto Prático 02
Parte 2: Matrizes

Clayton S. Rocha, Lucas G. Santana, Mateus M. Leal.

2194201 - 2088924 - 2126702

Departamento de Informática – Universidade Tecnológica Federal do Paraná (UTFPR)
Av. Sete de Setembro, 3165 – 80280-901 – Curitiba – PR – Brasil

Bacharelado em Sistemas de Informação – Universidade Tecnológica Federal do Paraná
Curitiba, PR.

Resumo. *Relatório sobre o Projeto prático 02, parte 02: Matrizes. O mesmo contém detalhes de como foram feitas as resoluções para os problemas apresentados, descrevendo o funcionamento das funções utilizadas para elaboração do projeto.*

1. Brainstorming

Inicialmente nos reunimos em sala de aula para elaboração de rascunhos de possíveis soluções para o algoritmo, e sem muitas dificuldades conseguimos um escopo do que deveria ser feito para que o algoritmo fosse eficaz. Começamos analisando e contando a vizinhança dá uma célula qualquer, e a partir de então começamos a projetar o algoritmo.

A primeira parte posta em prática, foi a criação de um menu interativo para o usuário, onde disponibiliza as opções, assim fazendo com que o usuário escolha o modo de jogo que deseja iniciar, lembrando que se o usuário informar uma opção fora das elencadas o jogo nunca se iniciara. Posterior ao criarmos o menu, demos início a criação de uma simples função cujo objetivo era esboçar o Jogo da Vida na tela do usuário.

Começamos a elaborar as regras em sala de aula, porém essa parte nos demandou mais tempo que as demais, foi onde surgiu grande parte dos nossos desafios em relação ao projeto. A continuidade foi realizada através de vídeo conferências, onde por fim conseguimos aplicar as regras que foram informadas no artigo, cedido pelos professores de maneira coerente e adequada para que o jogo funcionasse corretamente.

Considerando que alguns comandos são restritos ao Sistema Operacional Linux, uma das maiores dificuldades encontradas para a elaboração do projeto “Jogo da Vida”, foi a incompatibilidade de comandos no Sistema Operacional Windows, como por exemplo a função “sleep” e a função “system(‘clear’)”. Após montarmos o programa, ocorreram alguns erros ao tentarmos compilar, onde perdemos muito tempo procurando erros de sintaxe e até mesmo erros de lógica, quando na verdade era o simples fato de serem comandos restritos a determinados Sistemas Operacionais.

Apesar de ter ocorrido o episódio com as funções “sleep” e “system(‘clear’)” não encontramos maiores dificuldades para a conclusão do jogo proposto.

2. Int main()

Nossa main não é tão extensa, nela criamos o menu interativo destinada ao usuário. Para prevenir que o usuário faça uma escolha que não esteja elencada, utilizamos um While que enquanto a entrada do “sncanf” estiver fora do intervalo definido, ele não ira sair do menu até que o mesmo escolha uma opção dentre as elencadas.

Após verificar a escolha do usuário e saindo do laço, nós demos início a primeira matriz do program, uma matriz 10 x 10 e a preenchemos toda com zeros, assim sendo a criação de um tipo de “tabuleiro”.

Inicia-se a primeira chamada de função da nossa main, a função “criaImagem(matriz, escolha)”, onde passamos por parâmetro a matriz já criada e a opção escolhida pelo usuário, nessa função é onde ocorre a criação da primeira imagem a partir de várias verificações com “if” e “else if”, onde para cada escolha possível, dentro do intervalo proposto no menu (de “a” à “f” ou “A” à “F”) a uma criação de imagem distinta de acordo cm a escolha passada por parâmetro.

A função “printaMatriz(matriz)” serve para esboçar na tela o “tabuleiro” criado originalmente sem nenhuma alteração das regras, percorremos toda matriz em um laço de repetição “for”, dentro do mesmo, verificamos se na coordenada x,y o valor é igual a um, se sim “printamos” um zero (0), caso contrário “printamos” um simples ponto (.). Em seguida há a chamada da função “aplicaRegras” é nela em que o nosso jogo ganha vida. Onde preenchemos uma matriz cópia com zeros e em seguida percorremos a matriz original para aplicar as regras de Conway, utilizando “if” e “else if” verificamos se aquela célula possui o número suficiente de vizinhos para poder sobreviver, e aplicamos a regra na matriz cópia, em seguida atualizamos a matriz original, recebendo a matriz cópia assim fazendo a atualização para a próxima geração da matriz. Dentro da própria função “aplicaRegras” nós chamamos novamente a própria função assim fazendo com que o programa entre em um laço infinito.