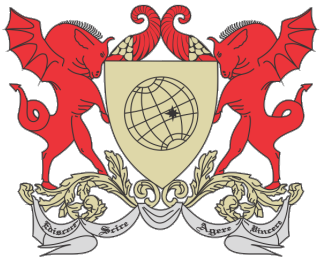

UNIVERSIDADE FEDERAL DE VIÇOSA
CAMPUS FLORESTAL

Trabalho Prático
Projeto e Análise de Algoritmos

Luciano Belo - 3897
Mariana Souza - 3898

Trabalho Prático apresentado à disciplina de Projeto e Análise de Algoritmos do curso de Ciência da Computação da Universidade Federal de Viçosa.

Florestal
Dezembro de 2021

CCF 330 - Projeto e Análise de Algoritmos

TP 00 - Gerador de Artes

Luciano Belo - 3897
Mariana Souza - 3898

14 de Dezembro de 2021

Contents

1 Introdução

O presente trabalho consiste na implementação de um gerador de artes, onde essas artes geradas pelo programa estarão em um quadro que contém 20 linhas e 80 colunas delimitadas por "|" em suas colunas e '-' nas linhas, além disso o programa também conta com um menu onde consta cinco opções, sendo que o usuário poderá escolher apenas uma das três, uma mistrura aleatória das três ou uma obra de arte que foi criada pelo aluno. Para o desenvolvimento do trabalho prático presente foi utilizado o VScode para implementação do código e o GitHub para o versionamento.

2 Desenvolvimento

O código foi desenvolvido na linguaguem C e está dividio em pastas, onde "src" é encontrado os arquivos .c e "headres" os arquivos .h, sendo que as funções para o quadro e implementação do algoritmo para apresentar as figuras estão no arquivo "quadro.c" já o arquivo para apresentar a interface para o usuário estão no "menu.c", contendo os cases para cada tipo de opção que for selecionada. Sendo assim, o trabalho segue a seguinte estrutura de pastas:

```
TP0-PAA
├── doc
├── headers
│   ├── menu.h
│   └── quadro.h
├── out
│   └── exec
└── src
    ├── main.c
    ├── menu.c
    └── quadro.c
```

2.1 Quadro

Para criar o quadro que foi utilizado para exibir as obras de artes foi criada uma matriz onde foram utilizados duas estruturas de repetição, assim preenchendo as bordas da matriz com "|" e as linhas laterais supeiores e inferiores com '-'. Foram criadas as funções "inicioQuadro", "CriaQuadro" e "printframe", onde em ordem primeiro temos a criação de uma alocação dinâmica para inciar um quadro e posteriormente uma função que cria o quadro com os tamanhos que foram especificados e para apresentar o quadro no prompt de comando.

2.2 Asterisco/Soma/LetraX

Nas três opções iniciais, para Asterisco, soma e Letra X foram utilizadas lógicas parecidas onde é usado rand() para inserir as figuras em posições aleatórias e também foi implementado na matriz a figura especifica para cada opção selecionada, uma iteração até a quantidade de figuras escolhidas é gerada, como aprsentado na Figura 1 abaixo:

```
void Asterisco(char **matriz, int quantidade)
{

    int x, y;

    for (int i = 0; i < quantidade; ++i)
    {
        while (1)
        {

            x = 1 + rand() % 18;
            y = 1 + rand() % 78;

            if (matriz[x][y] == ' ')
            {
                break;
            }
        }
        matriz[x][y] = '*';
```

Figure 1: Função gera obra de arte Asterisco.

Além das funções principais de criação das obras de arte temos a função "verificaEntrada" que está presente no arquivo quadro.c, contendo a funcionalidade de verificar as exceções que foram especificadas na descrição do presente trabalho prático, assim essa função verifica se o número e maior que 100 e caso seja arfimativo irá ser retornado o valor 100 e se a quantidade for menor ou igual a zero é gerado uma quantidade de figuras aleatórias no intervalo de 1 a 100.

2.3 Obra de Arte

A obra de arte escolhida pela dupla foi um conjunto de emoticon japonês, que também saõ conhecidos uma combinações de caracteres de pontuação da escrita japonesa que formam desenhos de rostos e expressões. Para a implementação dos emoticons foi necessário armazenar cada caractere que a linguagem conhece em cada espaço da matriz, foram realizados testes e percebemos que essa seria a única alternativa para representação na linguagem C.

Em cada iteração da opção 5 temos uma geração aleatória e difeerente de emoticons, onde foram geradas funções para 12 emojis diferentes, sendo assim são apresentadas emojis de quantidades diferentes e tipos diferentes. Os 12 emojis criados na obra de arte estão apresentados na tabela 2.

~(^-^)~	(^_^)	<(^.^)>	0_o
<(^^<)	<(o_o<)	<(-'.'-)>	<('.'-^)
(>';..;')>	(-_-;)	(^o^;)	(#^.^#)

Figure 2: Tabela de emoticons japoneses.

Foram criadas doze função para cada tipo de emoji divididas em "insert" que tem o objetivo de implementar cada parte do emoji armazenando nas posições da matriz, como apresentado na Figura 3 abaixo e a função "emotion", apresentada na Figura 4, que contém a função insert para realizar a posição aleatória do emoji.

```
void insertEmotionOne(char **frame, int x, int y)
{

    /*~(^-^)~*/
    frame[x][y] = '~';
    frame[x][y + 1] = '(';
    frame[x][y + 2] = '^';
    frame[x][y + 3] = '-';
    frame[x][y + 4] = '^';
    frame[x][y + 5] = ')';
    frame[x][y + 6] = '~';
};
```

Figure 3: Função insertEmoticonOne.

```
void emotionOne(char **frame, int quantidade)
{
    int x, y;
    srand(time(0));

    for (int i = 0; i < quantidade; ++i)
    {
        while (1)
        {
            x = 1 + rand() % 17;
            y = 1 + rand() % 77;

            if (frame[x][y] == ' ' && frame[x][y + 1] == ' ' && frame[x][y + 2] == ' '
            && frame[x][y + 3] == ' ' && frame[x][y + 4] == ' '
            && frame[x][y + 5] == ' ' && frame[x][y + 6] == ' ')
            {
                break;
            }
        }
        insertEmotionOne(frame, x, y);
```

Figure 4: Função emotionOne.

Abaixo está sendo apresentada a execução da opção 5 na Figura 5, mostrando os emojis que foram gerados aleatoriamente dentre os 12 disponíveis que estão representados na tabela, para serem apresentados na execução.

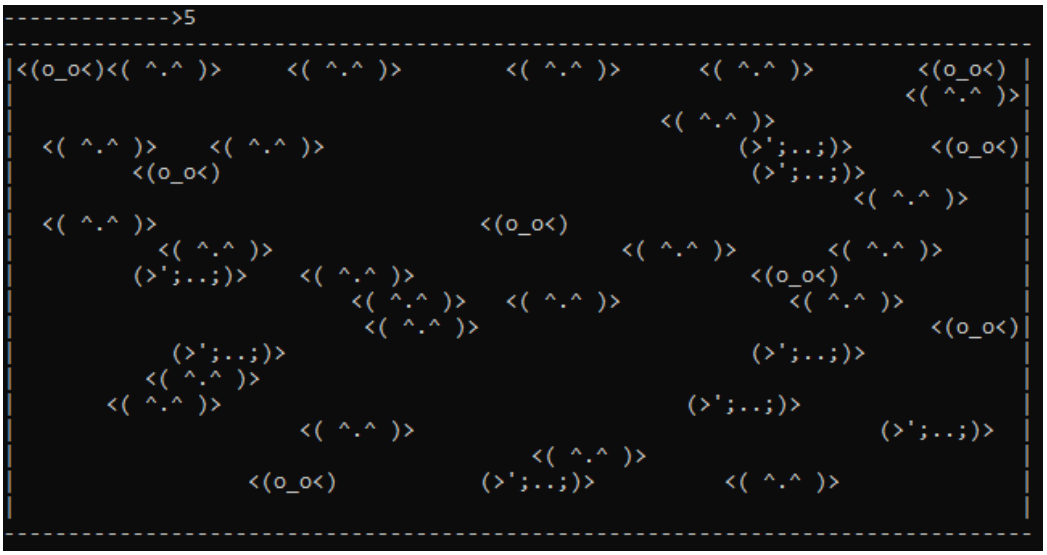

Figure 5: Saída opção 5.

3 Execução

O trabalho possui um arquivo makefile contendo um conjunto de diretivas usadas para automação de compilação, execução e remoção de arquivos binários e serão apresentados em seguida.

```
all:
	gcc src/main.c src/quadro.c src/menu.c -o out/exec
run:
	out/exec
clean:
	rm out/exec
clean_exec:
	rm out/exec.exe
```

Os resultados das execuções do programa estão sendo apresentados abaixo, onde são apresentados o menu na Figura 6 contendo as opções para criação das obras de arte, a execução da opção 4 gerando as 20 figuras aleatórias na Figura 7.

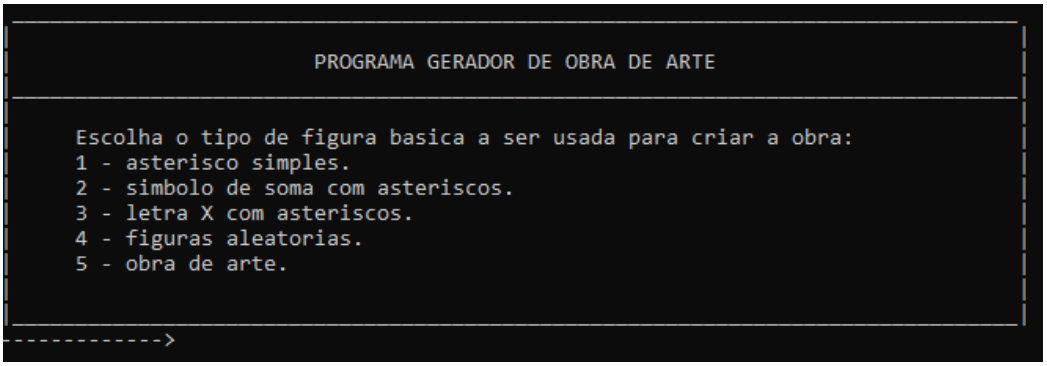

Figure 6: Menu.

Figure 7: Opção 4 figuras aleatórias.

Também estão sendo apresentadas abaixo as imagens da execução do gerador de obra de arte para o asterisco simples na Figura 8, o símbolo da soma com asterisco na Figura 9 e a Letra X com asterisco na Figura 10.

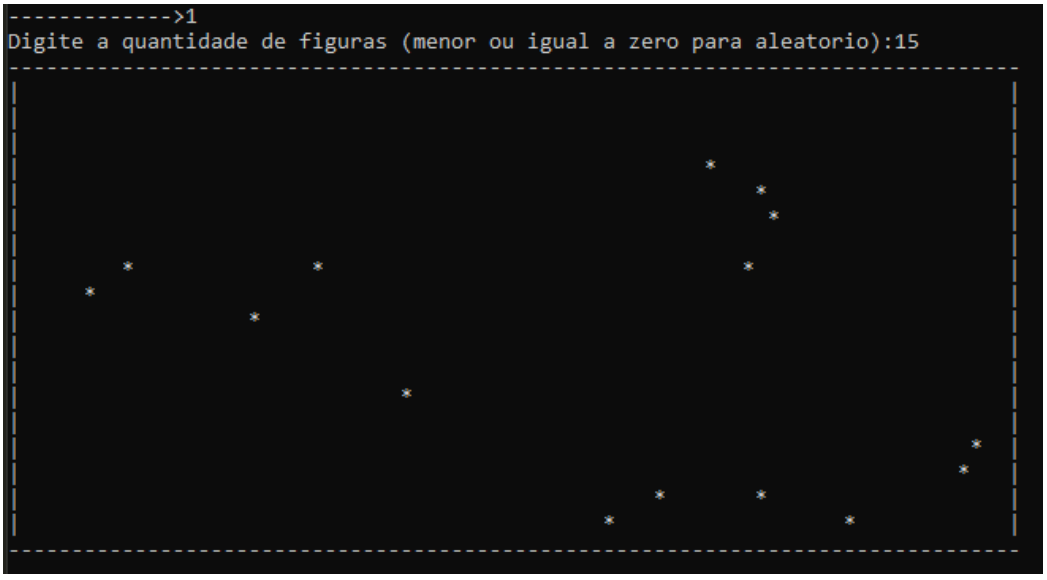

Figure 8: Asterisco Simples.

Figure 9: Soma com asterisco.

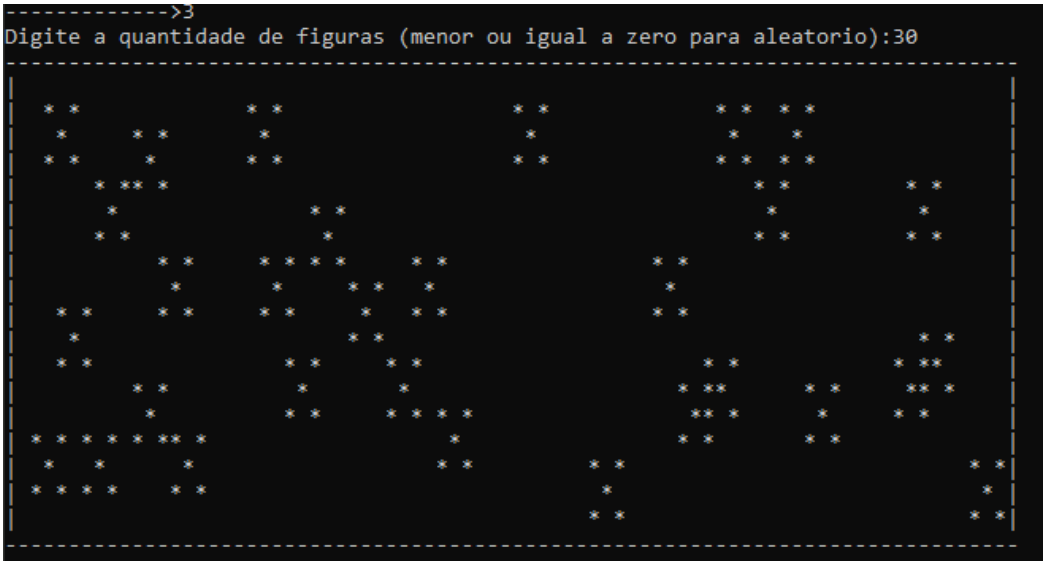

Figure 10: Letra X.

4 Conclusão

Desta forma, podemos concluir que o trabalho em questão foi desenvolvido conforme o esperado, atingindo todas as especificações requeridas na descrição do mesmo, já que o intuito principal do trabalho foi atingido, a criação de um programa para gerar obras de arte aleatórias.

Tivemos algumas dificuldades para a elaboração da obra de arte criada pelo o aluno, visto que, a lingugem C não permite alguns caracteres que gostariamos de ter colocado na elaboração dos emoticons, porém posterioemente conseguimos compreeender melhor como a linguagem funcioana e entregar a opção funcionando.

Posto isso, é válido dizer que apesar das dificuldades na implementação do código, a dupla foi capaz de superar e corrigir quaisquer erros no desenvolvimento dos algoritmos. Haja vista que o trabalho foi executado conforme o planejado, sendo tratado tudo que foi pedido pelo professor. Por fim, verificou-se a assertiva para o objetivo do projeto em implementar um programa para gerar obras de arte aleatórias.

References

[1] https://whimsical.com/tp0-opcao4-3CiZCfSyfo3vpDTdik6geh, OPÇÃO 4-WHIMSICAL, Acesso em 14 de Dezembro de 2021;

[2] https://www.dicionariopopular.com/kaomoji-emojis-carinhas-kawaii/, KAOMOJI OS EMOJIS JAPONESES, Acesso em 12 de Dezembro de 2021;