Ramo Estudantil IEEE - UEL

Arthur Henrique P Correa (arthur.pozzobon137@uel.br)
Marco Antonio Henry (marco.antonio.henry@uel.br)
Vitor Luiz dos Santos (Vitor.Luiz.Santos@uel.br)
Pedro Antony Dias (pedro.antony.dias@uel.br)

RELATÓRIO FINAL:

Código Morse

Londrina
2023

Ramo Estudantil IEEE - UEL

Arthur Henrique Pozzobon Correa
Marco Antonio Cottorello Henry
Vitor Luiz dos Santos
Pedro Antony dias

RELATÓRIO FINAL:

Projeto Código Morse

Relatório apresentado ao Ramo Estudantil IEEE da Universidade Estadual de Londrina.

Diretor de Projetos: Nathan Andreani Netzel
Gestores de Projetos: Daniel Tresse Dourado, Levi Monteiro dos Santos

Londrina
2023

Contato do Ramo: sb.uel@ieee.org
Institute of Electrical and Electronics Engineers – IEEE
Universidade Estadual de Londrina - UEL • Paraná - Brasil

CORREA, Arthur Henrique Pozzobon. HENRY, Marco Antonio Cottorello. SANTOS, Vitor Luiz. DIAS, Pedro Antony. **Relatório Final**: Projeto Código Morse. 2023. 17 folhas. Relatório apresentado ao Ramo Estudantil IEEE da Universidade Estadual de Londrina, Londrina, 2023.

RESUMO

O projeto Código Morse consiste em criar um codificador do alfabeto padrão para o sistema de códigos Morse e um decodificador do mesmo de volta para a linguagem padrão. Esse código utilizará a linguagem C de programação, com suas bibliotecas e funções.

SUMÁRIO

Introdução a Linguagem C

Objetivo

Com este projeto o objetivo foi de aprender mais sobre a programação relacionado aos projetos da Faculdade e do IEEE criando um código que é capaz de ler codificando um texto Morse em Alfanumérico em vice-versa.

1. Introdução a Linguagem C

A linguagem C foi desenvolvida para ser uma sucessora da linguagem B, criada em 1970 nos laboratórios AT&T criada por Ken Thompson. Ela foi criada com o objetivo de ser uma linguagem voltada a sistemas operacionais, compiladores e outras ferramentas de baixo nível graças a sua simplicidade.

1. FUNÇÕES UTILIZADAS

STRING.H

É uma biblioteca que provém o uso de comandos que facilitam a manipulação de strings, vetores usados para armazenamento de caracteres, que são um tipo de "memória" de ordem e natureza do elemento. Ou seja, ao salvar algum ideal que deseja utilizar posteriormente e não quer reescrevê-lo criamos uma função "VOID", onde será salvo o ideal que o usuário queira passar e poderá ser chamado a hora que necessitar.

Ctype.h

Uma função que compreende se uma palavra digitada está escrita em Maiúscula, minúscula, espaços vazios, dígitos ou caracteres especiais.

Stdio.h

A biblioteca padrão da linguagem C (ou C++) oferece funcionalidade de saídas e entradas padrão, em outras palavras, leitura de dados a partir do teclado e a exibição.

Função Char

O tipo char serve apenas e exclusivamente para armazenar UM carácter para ser declarado.

Função Strtok

Serve para quebrar uma String em partes menores com bases em um conjunto de delimitadores específicos.

Vetores

É uma estrutura que armazena uma coleção de elementos de um mesmo tipo em único nome.

Const Char

Diferente do char, o const char ele cria uma função que não pode ser alterada de forma nenhuma, ou seja, tudo que você escrever dentro do const char é armazenado e poderá ser utilizado posteriormente, porém, porém não poderá criar nada de diferente dentro da função.

Função Toupper

Função vinda da biblioteca “Ctype.h” onde ela identifica se a letra digitada é MAIÚSCULA ou minúscula.

Função STRCMP

Função vinda da biblioteca “String.h” que compara duas strings diferentes.

Getchar

Usada para ler um carácter (um byte) padrão do teclado, além de ser uma função que faz parte do ‘Int” que gera o retorno de valor inteiro.

Fgets

Função padrão da biblioteca “stdio.h” usada na linguagem C para ler texto.

Sizeof

Um operador que retorna um tamanho em byte de um tipo de dado ou expressão. Ela é muito utilizada para descobrir a quantidade de variáveis que foram utilizadas em determinado texto.

Stdin

É um ponteiro que apresenta entrada padrão apresentada pelo teclado.

Break

Usada para encerrar um processo de estrutura de repetição (loop) ou seleção.

funções void

É uma função que não retorna a nenhum valor. Em outras palavras, não existe nenhum tipo de retorno vazio.

IF E ELSE

São controle de fluxo na linguagem C para gerar decisões condicionais com base em um valor Booleano.

IF: Caso a condição seja verdadeira e satisfaz a condição requerida;
ELSE: Caso a condição IF seja falsa e não satisfaz a condição verdadeira passará para a função ELSE.

Printf

Usado para imprimir um texto ou uma mensagem que usuário terá que seguir.

Scanf

Função que irá interpretar dados formatados a ele a partir da entrada padrão.

While

Uma estrutura de controle de fluxo em C, usada para criação de loops ou repetições baseadas em uma condição

For

Estrutura de controle de fluxo em C, diferentemente do while/do while, que cria uma estrutura de repetição infinito, o FOR é mais restrito sobre, permitindo que essa estrutura você possa manipulá-la da forma que bem desejar para gerar um loop FINITO.

Int

É um dado fundamental que permite a leitura de valores inteiros.

Int Main()

É a função principal que é chamada quando o programa se inicializa. É o processo de entrada para todo possível código fonte.

Array

Um array é uma estrutura de dados que armazena uma coleção de elementos do mesmo tipo. Os elementos são acessados por um indicativo da posição dentro de um array.

2 Construção Morse

2.1 Inserindo as Bibliotecas

Para iniciar o código, introduziremos as bibliotecas, que possuem as funções que vamos utilizar.

Figura 1: Bibliotecas.

```c
#include <stdio.h>// Biblioteca base para os comandos em linguagem C.
#include <string.h>// Biblioteca que possibilita manipular melhor frases em linguagem C.
#include <ctype.h> // Biblioteca que facilita manipular caracteres na linguagem C.
```

Fonte: Screenshot do arquivo .c no VSCode

2.2 Início da função de codificar

Nessa etapa, foi criado a função que será responsável por codificar uma string em linguagem padrão para código Morse. Para isso, primeiramente, foi colocado um array com os números e letras do alfabeto traduzidos para código Morse.

Figura 2: Array com as letras e números traduzidos.

```c
// Função para codificar.
void codificarCaractere(char c) {
    const char *morse[] = { //array de strings.
        ".-", "-...", "-.-.", "-..", ".", "..-.", "--.", "....", "..", ".---",
        "-.-", ".-..", "--", "-.", "---", ".--.", "--.-", ".-.", "...", "-",
        "..-", "...-", ".--", "-..-", "-.--", "--..",   // Caracteres em morse.
        "-----", ".----", "..---", "...--", "....-", ".....", "-....", "--...",
        "---..", "----."    //Numeros em morse.
    };
```

Fonte: Screenshot do arquivo .c no VSCode.

Em seguida, ainda dentro da função de codificar, colocaremos uma série de condições em sequência, que serão responsáveis por analisar o caractere de entrada feito pelo usuário e buscar seu correspondente. Ele poderá ser uma letra, um número ou um espaço. Logo após, foi fechado a função codificar.

Figura 3: Condições.

```
15      if (c >= 'A' && c <= 'Z') { //If para verificar o caractere e colocar seu correspondente.
16          printf("%s ", morse[c - 'A']);
17      } else if (c >= '0' && c <= '9') { //If para analisar se for um número.
18          printf("%s ", morse[c - '0' + 26]);
19      } else if (c == ' ') { // If quando o caractere for espaço.
20          printf(" / "); // Imp: na hora de printar na tela o espaço, colocamos uma barra entre os espaços.
21      }
22
23  }
```

Fonte: Screenshot do arquivo .c no VSCode.

2.3 inícios da função decodificar

No início da função decodificar é criado dois arrays. O primeiro com os caracteres em código Morse, já o segundo com os correspondentes do Morse em alfanumérico.

Figura 4: Decodificador.

```
25  // Função para decodificar
26  void decodificarcaractere(const char *codigoMorse) {
27      const char *morse[] = {
28          ".-", "-...", "-.-.", "-..", ".", "..-.", "--.", "....", "..", ".---",
29          "-.-", ".-..", "--", "-.", "---", ".--.", "--.-", ".-.", "...", "-",
30          "..-", "...-", ".--", "-..-", "-.--", "--..", // Caracteres em morse.
31          "-----", ".----", "..---", "...--", "....-", ".....", "-....", "--...",
32          "---..", "----.","/" // Numeros em morse.
33      };
34      const char *alfanumerico[] = { // Correspondentes do morse em alfanumerico.
35          "A", "B", "C", "D", "E", "F", "G", "H", "I", "J",
36          "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T",
37          "U", "V", "W", "X", "Y", "Z",
38          "0", "1", "2", "3", "4", "5", "6", "7", "8", "9"," "
39
40          };
```

Fonte: Screenshot do arquivo .c no VSCode.

Após isso, é realizado uma quebra da string em subtokens (caractere em Morse) que irá parar a leitura quando ele detectar um espaço. É criado um loop para ler toda essa sequência e buscar o caractere correspondente do Morse em alfanumérico e coloca ele na tela do usuário. Fim da função decodificar.

Figura 5: detector de caracteres vazios.

```
42      char *token = strtok((char *)codigoMorse, " "); // Divide a string em subtokens e para quando detecta um espaço(delimitador).
43
44      while (token != NULL) { // Enquanto nao chegar no \0.
45          for (int i = 0; i < 37; i++) { // Percorre a sequencia de caracteres em morse.
46              if (strcmp(token, morse[i]) == 0) { // Busca o caractere correspondente.
47                  printf("%s", alfanumerico[i]);
48                  break;
49              }
50          }// Fim do loop.
51
52          token = strtok(NULL, " "); // Busca o proximo token.
53
54      }
```

Fonte: Screenshot do arquivo .c no VSCode.

2.4 Função main

Inicialmente o usuário terá de escolher entre duas opções, codificar ou decodificar, caso não escolha nenhuma das duas, será mostrado para ele que a opção é inválida. Após feita a escolha, o código irá pedir dentro do “PRINTF” o que deseja ser traduzido, feito isso passará pela função “FGETS”, que vai receber a mensagem escrita pelo usuário. Com esse processo feito, a partir de um loop “FOR”, a função competente à situação será “chamada” e mostrará na tela do usuário o texto codificado ou decodificado conforme a escolha inicial.

Figura 6: Interface com o usuário.

```c
int main() {
    char mensagem[256]; // Maximo de 256 caracteres.
    int escolha;

    printf("Escolha:\n");     // Escolha se quer codificar ou decodificar.
    printf("1. Codificar em codigo Morse\n");
    printf("2. Decodificar codigo Morse\n");
    scanf("%d", &escolha);
    getchar(); // Limpar a entrada.

    if (escolha == 1) {
        printf("Digite uma mensagem para codificar em codigo Morse: ");
        fgets(mensagem, sizeof(mensagem), stdin);
        mensagem[strlen(mensagem) - 1] = '\0';  // Remove o caractere de nova linha.

        printf("Mensagem codificada em codigo Morse: ");
        for (int i = 0; mensagem[i]; i++) {     //Chama a funcao de decodificar em um loop.
            codificarCaractere(toupper(mensagem[i]));// Deixa a mensagem em maiusculo e printa ela codificada.
        }
        printf("\n");
    } else if (escolha == 2) {
        printf("Seguindo como este exemplo: (... / --- / ... )\n");
        printf("Digite um codigo Morse (com espaco e uma barra entre as palavras) para decodificar: ");
        fgets(mensagem, sizeof(mensagem), stdin);
        mensagem[strlen(mensagem) - 1] = '\0';  // Remove o caractere de nova linha.

        printf("Mensagem decodificada: ");
        decodificarcaractere(mensagem); // Chama a função e printa a mensagem decodificada.
    } else {
        printf("Opçao invalida.\n");
    }

    return 0;
}
```

Fonte: Screenshot do arquivo .c no VSCode.

3. VSCode

O visual Code (também conhecido como VSCode) é um editor de código fontes gratuito e de código aberto pela Microsoft. Graças a sua ampla gama de extensões suportes, o tornou-o extremamente popular.

Uma das características mais marcanes do VSCode é a sua ampla compatibilidade com várias linguafens de programação e tecnologia. Ela suporta uma vasta gama de linguagens, desde as mais populares até das mais específicas.

Além disso, o VSCode é altamente extensível. A comunidade de desenvolvedores contribui ativamente com uma vasta gama de extensões que ampliam suas funcionalidades. Essas extensões vão desde suporte de linguagem específicas até ferramentas de integração com serviços de nuvem e ambientes de desenvolvimento.

Em resumo, o Visial Studio Code se destaca por ser um editor altamente flexível, eficiente, compatível com ampla gama de tecnologias e linguagens de programação.

4. Compiladores

Os compiladores são programas de software que traduzem o código-fonte escrito em uma linguagem de programação de alto nível (como C, C++, Java Script, Python) para uma forma que possa ser executado diretamente pelo computador, chamada de código de máquina ou código binário. Eles desempenham um papel crucial no desenvolvimento de software, permitindo que os programadores escrevam código em linguagens de alto nível que são mais compreensíveis e eficientes do que o código de máquina.

5. RESULTADOS E DISCUSSÕES

Com a montagem feita – para realizar o teste do funcionamento do código em C++, foi utilizado o VSCode, onde deixamos as funções separadas acima do código para poder chamá-las eventualmente.

CONCLUSÕES

Com os testes podemos concluir que o codificador e o decodificador foi um sucesso na hora do teste, executando a codificação com sucesso de alfabeto para código Morse e o contrário também.

Figura 7: Resultado da codificação de alfanumérico para Morse.

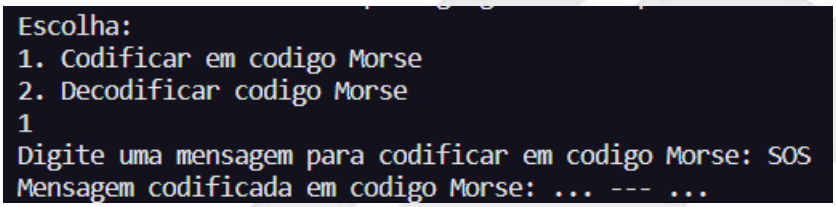
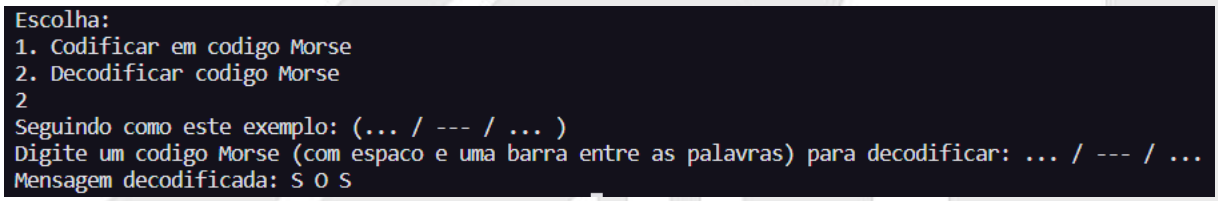


Figura 8: Resultado da decodificação de Morse para alfanumérico



Fonte: Screenshot do arquivo .c no VSCode.

REFERÊNCIAS BIBLIOGRÁFICAS

Site do funcionamento da Linguagem C: https://petbcc.ufscar.br/string/. Acesso dia 24/09/2023.

Como funciona o Código Morse: https://www.todamateria.com.br/o-que-e-codigo-morse/. Acesso dia 24/09/2023.

O que é o VSCode? https://www.treinaweb.com.br/blog/vs-code-o-que-e-e-por-que-voce-deve-usar. Acesso dia 26/09/2023.

O que é um Compilado https://www.youtube.com/watch?v=afUiVvDUIRA. Acesso dia 26/09/2023.