UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
INSTITUTO METRÓPOLE DIGITAL
GRADUAÇÃO EM TECNOLOGIA DA INFORMAÇÃO

RELATÓRIO TÉCNICO: Projeto de Introdução às Técnicas de Programação - Unidade 1

ANA CLARA LIMA DA SILVA

NATAL, RIO GRANDE DO NORTE
2025

Departamento de Informática e Matemática Aplicada

Introdução às Técnicas de Programação — IMD0012

NOME DO PROJETO: Conversor de unidades.

OBJETIVO: O projeto visa realizar a conversão entre unidades de maneira prática.

PROBLEMA SOLUCIONADO: Necessidade de consultar fórmulas e tabelas para converter valores em unidades de medida diferentes.

JUSTIFICATIVA: Este projeto foi escolhido visando facilitar a conversão de valores em unidades de medida diferentes, sem a necessidade de consulta de tabelas e utilização de fórmulas de conversão pelo usuário, visto que o programa já realiza a conversão.

METODOLOGIA

Compilador utilizado: GCC versão 15.2.0

Editor utilizado: Visual Studio Code

APLICAÇÃO DOS CONCEITOS DA UNIDADE 1

Como foram utilizadas as variáveis:

As variáveis foram utilizadas para armazenar as informações fornecidas pelo usuário, usadas nas funções e para serem armazenadas nos vetores, utilizados como histórico.

Como foram utilizadas as estruturas condicionais:

Na linha 18, se o total de conversões for igual a zero, será exibido na tela que nenhuma conversão foi realizada, observe abaixo:

```
18      if(total_conversoes == 0) {
19          printf("Nenhuma conversao realizada.\n");
20          return;
21      }
```

Já na linha 77, se a opção for diferente de 4, o programa retornará o menu principal, observe abaixo:

```
77          if (opcao != 4) {
78          printf("\nPressione Enter para continuar.");
79          getchar(); //Limpa o Enter do scanf anterior
80          getchar(); //Espera o usuário pressionar Enter
81          }
```

Nas funções de conversões, de acordo com a opção escolhida, a conversão entre unidades será realizada ou a entrada será considerada inválida. Observe as linhas 101, 111 e 121 na função *convertercomprimento*:

```
88  void convertercomprimento() {
101     if (escolha == 1) {
102         resultado = valor * 100;
103         printf("%.2f metros = %.2f centimetros\n", valor, resultado);
104 
105         //Adiciona ao vetor de histórico
106         historico_valores[total_conversoes] = valor;
107         historico_resultados[total_conversoes] = resultado;
108         strcpy(historico_tipos[total_conversoes], "m -> cm");
109         total_conversoes++;
110 
111     } else if (escolha == 2) {
112         resultado = valor / 100;
113         printf("%.2f centimetros = %.2f metros\n", valor, resultado);
114 
115         //Adiciona ao vetor de histórico
116         historico_valores[total_conversoes] = valor;
117         historico_resultados[total_conversoes] = resultado;
118         strcpy(historico_tipos[total_conversoes], "cm -> m");
119         total_conversoes++;
120 
121     } else {
122         printf("Opcao invalida.\n");
123     }
124 }
```

Observe também as linhas 139, 149 e 159 na função *convertertemperatura*:

```c
126  void convertertemperatura() {
139      if (escolha == 1) {
140          resultado = (valor * 9 / 5) + 32;
141          printf("%.2f°C = %.2f°F\n", valor, resultado);
142
143          //Adiciona ao vetor de histórico
144          historico_valores[total_conversoes] = valor;
145          historico_resultados[total_conversoes] = resultado;
146          strcpy(historico_tipos[total_conversoes], "°C -> °F");
147          total_conversoes++;
148
149      } else if (escolha == 2) {
150          resultado = (valor - 32) * 5 / 9;
151          printf("%.2f°F = %.2f°C\n", valor, resultado);
152
153          //Adiciona ao vetor de histórico
154          historico_valores[total_conversoes] = valor;
155          historico_resultados[total_conversoes] = resultado;
156          strcpy(historico_tipos[total_conversoes], "°F -> °C");
157          total_conversoes++;
158
159      } else {
160          printf("Opcao invalida.\n");
161      }
162  }
```

A lógica das estruturas de repetição implementadas:

Na linha 23, percorre vetor do histórico, observe:

```c
15   void mostrar_historico() {
23       //Repetição: Percorre vetor do histórico
24       for(int i = 0; i < total_conversoes; i++) {
25           printf("%d. %s | Entrada: %.2f | Saida: %.2f\n",
26                  i + 1,
27                  historico_tipos[i],
28                  historico_valores[i],
29                  historico_resultados[i]);
30       }
31   }
```

Na linha 50, mostra o menu usando vetor, observe:

```c
33   int main() {
46       do {
49           //Repetição: Mostra menu usando vetor
50           for(int i = 0; i < 4; i++) {
51               printf("%d. %s\n", i + 1, menu_opcoes[i]);
52           }
```

Na linha 57, o switch-case executa diferentes ações baseada em condições, nesse caso são as opções escolhidas pelo usuário, observe:

```
33  int main() {
46      do {
53          //Lê a escolha do usuário:
54          scanf("%d", &opcao);
55
56          //Switch-case utilizado para direcionar as opções
57          switch (opcao)
58          {
59              case 1: //Se a opção for 1, o código abaixo será executado
60                  printf("Voce escolheu Comprimento!\n");
61                  convertercomprimento(); //Chama a função para converter comprimento
62                  break; //Sai do switch
63              case 2:
64                  printf("Voce escolheu Temperatura!\n");
65                  convertertemperatura(); //Chama a função para converter temperatura
66                  break;
67              case 3:
68                  mostrar_historico();
69                  break;
70              case 4:
71                  printf("Saindo...\n");
72                  break;
73              default: //Se nenhum dos cases anteriores for executado, ele será
74                  printf("Opcao invalida!\n");
75          }
```

Na linha 83 irá repetir o menu até o usuário escolher "sair", observe:

```
33  int main() {
46      do {
83      } while(opcao != 4); //Irá repetir o menu até o usuário escolher "sair"
```

Como os vetores foram aplicados no projeto:

Foram usados para armazenar o histórico dos valores de entrada, resultados, os tipos e o menu de opções em string.

Funções criadas:

Além da função *main*, foram criadas também as funções: *convertercomprimento*, que converte o comprimento, de Metros para Centímetros e de Centímetros para Metros; *convertetemperatura*, que converte a temperatura, de Celsius para Fahrenheit e de Fahrenheit para Celsius; *mostrar_historico*, que apresenta o histórico de conversões solicitadas pelo usuário.

ESTRUTURA DE DADOS

Vetores utilizados:

- *historico_valores[10]*: Vetor para valores de entrada;
- *historico_resultados[10]*: Vetor para resultados;
- *historico_tipos[10][50]*: Vetor de strings para tipos;
- *menu_opcoes[]*: Vetor de strings para menu.

Variáveis utilizadas:

- *total_conversoes = 0* : Contador
- *opcao*: Variável utilizada para armazenar a escolha do usuário;
- *escolha*: Variável que armazena a escolha de conversão do usuário;
- *valor*: Variável que armazena o valor que o usuário deseja converter;
- *resultado*: resultado da conversão.

DIFICULDADES ENCONTRADAS

O VS Code apresentou algumas falhas no terminal.

SOLUÇÕES IMPLEMENTADAS

Provisoriamente as falhas foram corrigidas e depois retornaram, então não foram encontradas soluções adequadas.

ORGANIZAÇÃO DO CÓDIGO

O código foi organizado com funções que realizam as conversões e mostra o histórico, vetores que armazenam os valores de entrada fornecidos pelo usuário e a saída.

CONCLUSÃO

A maneira a qual foi feito o código está apropriada, porém, há a necessidade de serem implementadas conversões entre muitas outras unidades, visto que á somente Metros e Centímetros e Celsius e Fahrenheit.