

# SISTEMA PARA CLASSIFICAÇÃO DE EQUIPES DO EVENTO DESCIDA DA LADEIRA

LUIGI CORDEIRO DE OLIVEIRA - [luigi.oliveira01@fatec.sp.gov.br](mailto:luigi.oliveira01@fatec.sp.gov.br)

LUIS MATEUS SANTOS DE OLIVEIRA - [luis.oliveira107@fatec.sp.gov.br](mailto:luis.oliveira107@fatec.sp.gov.br)

LUIS GUSTAVO CASTIGLIONI - [luis.raimundo@fatec.sp.gov.br](mailto:luis.raimundo@fatec.sp.gov.br)

PEDRO CHRISTENSEN NOBRE - [pedro.nobre@fatec.sp.gov.br](mailto:pedro.nobre@fatec.sp.gov.br)

ROGERIO ANTONIO AUGUSTO - [rogerio.augusto@fatec.sp.gov.br](mailto:rogerio.augusto@fatec.sp.gov.br)

## RESUMO

Para podermos testar e aprender sobre a linguagem C foi desenvolvido esse projeto com a necessidade de um meio de medição de tempo para a descida da ladeira na Fatec, juntando os vários conhecimentos aprendidos no curso até o momento e a dedicação de todos os membros fizemos um programa simples, prático e intuitivo para o usuário, e cumprindo seu papel. O sistema tem as funções de coletar e armazenar tempos, gerar uma média, selecionar os melhores tempos e exportá-los para um arquivo TxT.

Palavras-Chave: Linguagem C; projeto integrador; sistema de gerenciamento.

## ABSTRACT

To test and Learn about the C programming language we developed this project with the necessity of a time counter to the downhill in Fatec, with the knowledge we have in the moment and the dedication of all members we build a simple, quick and intuitive system for the user completing his task. The software has the functions to collect and keep times, create an average time, select the best times and export them into a TxT file.

**Keywords:** C language; integrative project; management system.

## 1. INTRODUÇÃO

O carrinho de rolimã é feito artesanalmente, utilizando-se de materiais e ferramentas simples como madeira e rolamentos. Onde há uma base com rodas de bilha nas extremidades, um eixo móvel na parte frontal que é responsável pelos movimentos do carrinho e duas maçanetas nas laterais responsáveis pelos freios, onde não há um padrão para se fazer e muitos optam por uma junção de peso e aerodinâmica para atingir altas velocidades (RIOMEMORIAS, 2020).

A descida da ladeira é um evento promovido anualmente pela FATEC Arthur de Azevedo, onde várias equipes disputam numa corrida com carrinhos de rolimã, qual será o mais veloz e terá o melhor tempo. O evento se divide em duas categorias, *Speed* e Alegoria, sendo *Speed* apenas para quem irá competir pelos melhores tempos, e Alegoria para o pessoal que irá exibir seu carrinho. O dia conta também com uma banda e *food trucks* para alimentação (FATECMM, 2022).

O objetivo do projeto foi criar um programa na linguagem C que consiga receber os tempos dos carrinhos de Rolimã, salvar os menores e exportá-los para um novo arquivo, tendo em vista as medições de minutos, segundos e centésimos-de-segundos como o padrão para os tempos e deram a possibilidade de melhorar o software com outros conhecimentos aprendidos de forma independente.

## 2. FUNDAMENTAÇÃO TEÓRICA

A linguagem C foi criada por Dennis Ritchie nos laboratórios da *Bell Telephone* em 1972. C foi criada com um propósito: ser usada no desenvolvimento de uma nova versão do sistema operacional *Unix*. A primeira versão do *Unix* utilizava *Assembly*. Então podemos dizer que desde o princípio C foi uma linguagem criada por programadores para programadores (DAMAS, 2007).

A linguagem C é considerada de propósito geral, ou seja, é uma linguagem capaz de ser usada para praticamente qualquer tipo de projeto. É extremamente portátil, ou seja, um programa escrito em linguagem C pode ser facilmente utilizado em qualquer plataforma. Utilizando linguagem C podemos criar sistemas operacionais, aplicativos de todos os tipos, drivers e outros controladores de dispositivos, programar microcontroladores, etc. Além de toda essa flexibilidade, C é capaz de gerar programas extremamente rápidos em tempo de execução, possui uma sintaxe simples e poderosa, com instruções de alto nível (DAMAS, 2007).

A linguagem C influenciou de forma direta muitas linguagens como C++, Java, C#, *Objective C*, e muitas outras linguagens de programação tem sua sintaxe e estruturas influenciadas por C. C é uma linguagem extremamente popular e existem muitos compiladores C disponíveis para todas as plataformas (DAMAS, 2007).

Dentre os muitos comandos desta linguagem, podemos destacar o *void*, *switch case*, *While*, *if* e *for*. Vejamos suas funcionalidades e sintaxes (Damas, 2007):

- Função *Void* - Em inglês, *void* quer dizer vazio e é isto mesmo que o *void* é. Ele nos permite fazer funções que não retornam nada e funções que não têm parâmetros, como ela é uma função tem que ser declarada antes do *int main*, que também é outra função. A função *main* devolve um inteiro para informar o sistema operacional sobre o fim da execução do programa. A função devolve 0 para informar que o programa terminou de maneira normal e devolve um número diferente de 0 para informar que o programa terminou de maneira excepcional. (Veja também a função *exit*.)

Tipicamente, usam-se as constantes:

*EXIT\_SUCCESS* (que vale 0) ou *EXIT\_FAILURE* (que vale 1) como valor devolvido por *main* (Damas, 2007).

É um erro definir a função *main* como se ela fosse do tipo *void* pois:

```
void main (...) {retorna vazio.  
...  
}
```

- Comando *IF/ELSE* - é uma estrutura de decisão que examina uma ou mais condições e decide quais instruções serão executadas dependendo se a condição foi ou não foi. O comando *IF/ELSE* serve para alterar o fluxo de execução de um programa em C baseado no valor, verdadeiro ou falso, de uma expressão lógica. (Damas, 2007), A sua sintaxe é:

```
if (expr_log) {  
comando1; // executados se "expr_log" for verdadeira
```

```

    }
else
{
comando4;      //      executado      se      "expr_log"      for      falsa
}

```

- O comando *switch case* em C. É uma forma de reduzir a complexidade de vários *if ... else* encadeados. É muito utilizado, principalmente para uso em estruturas de menu. O conteúdo de uma variável é comparado com um valor constante, e caso a comparação seja verdadeira, um determinado comando é executado. (Damas, 2007), Sua sintaxe é:

```

switch          (variável          ou          valor)
{
case                                valor1:
case                                valor2:
case                                valor3:
    //                                código                                1
    break;
case                                valor4:
case                                valor5:
case                                valor6:
    //                                código                                2
    break}

```

- Estrutura de repetição *while* executa a repetição de um bloco de instruções enquanto uma condição é verdadeira. Pseudocódigo  
A estrutura enquanto ... faça equivale a estrutura *while* em linguagem C. (Damas, 2007), A sintaxe:

```

Iniciar a variável de controle
Enquanto (condição) faça
Início
Instruções;
Atualizar a variável de controle;
Fim;

```

- O comando *for* é utilizado, utilização do comando *for* O laço *for* é uma estrutura de repetição muito utilizada nos programas em C. É muito útil quando se sabe de antemão quantas vezes a repetição deverá ser executada. Este laço utiliza uma variável para controlar a contagem do *loop*, bem como seu incremento. (Damas, 2007), A sintaxe é:

```

for(valor_inicial;          condição_final;          valor_incremento)
{
instruções;
}

```

Dentro das muitas bibliotecas existentes dentro da linguagem C, podemos destacar: *locale.h*, *stdlib.h*, *stdio.h*, *windows.h*, *conio.h*, *stdbool.h*. Vejamos suas funcionalidades (Damas, 2007):

- *stdio.h* - é um cabeçalho da biblioteca padrão do C. Seu nome vem da expressão inglesa standard input-output header, que significa "cabeçalho padrão de entrada/saída".
- *Stdlib.h* - é um arquivo cabeçalho da biblioteca de propósito geral padrão da linguagem de programação C. Ela possui funções envolvendo alocação de memória, controle de processos, conversões e outras.
- *locale.h* - a utilização do arquivo *locale.h* e da função *setlocale()* configurada adequadamente vai garantir que caracteres como "ç" e acentuação sejam exibidos normalmente em nosso programa.
- *Windows.h* - contém declarações para todas as funções da API do Windows, todos os macros comuns utilizados pelos programadores do Windows.
- *conio.h* - é para desenhar tela, e é para dos/Windows (as funções do *conio* são úteis para manipular caracteres na tela, especificar cor de carácter e de fundo).
- *stdbool.h* - é um arquivo cabeçalho da biblioteca padrão da linguagem de programação C usada para manipular variáveis lógicas, como verdadeiro e falso. Pode ser substituída pela utilização de valores inteiro, sua função é simplesmente facilitar a compreensão do código.

### 3. MATERIAIS E MÉTODOS

O trabalho iniciou-se de forma simples e foi se desenvolvendo a cada dia e em grandes proporções. Com o passar do tempo, adquirimos mais ideias e as colocamos em prática de forma que o mesmo foi se desenvolvendo dia após dia. A princípio, era um programa bem rudimentar, porém, conforme fomos adquirindo novas ideias e novos aprendizados, ele foi tomando forma, assim pensamos em como formular a estrutura, o corpo do projeto, imaginando e testando cada tela e menu com suas opções e entradas necessárias.

O programa foi feito na IDE *CodeBlocks* sendo a Ide que estamos mais acostumados a programar utilizando a linguagem C.

Foi utilizado no software as Bibliotecas: *locale.h*, *stdlib.h*, *stdio.h*, *windows.h*, *conio.h*, *stdbool.h*.

Algumas das funções usadas foram: *printf*, *scanf*, *void*, *while*, *if*, *else*, *else if*, *bubble sort*, *default*, *switch case*, *sort*, *for*, *file*, *handle*. Foi criado uma função chamada *void* que mostrará o menu que dê para usar com as setas do teclado, e foi chamado de linha coluna.

Logo após, colocamos a função *set console*, que é a posição do cursor na posição, logo após fazer outra função, que é para colocar cor no texto e na tela, e como parâmetro foi colocado o valor das letras e o valor de fundo. Depois disso, foi criado um valor com o nome das cores do fundo e das letras e depois foi criada a caixa onde foi colocado o menu. Colocamos o nome de *void box*. E recebeu como parâmetros a linha 1, a coluna 1, a linha 2, a coluna 2, colocamos algumas variáveis, i, j tamanho da linha e da coluna para fazer

as interações, usamos o comando FOR, onde ele percorre da linha 1 até a coluna 12, ele o comando for vai colocando a função *void* linha coluna e preenchendo com a interação.

Foi criado o comando *while*, para entrar dentro do looping para receber as opções que o usuário estará selecionando.

Foi na função *void* que colocamos o nosso menu de opções, onde ficará impresso na tela o nome da equipe, o menu principal, as três baterias, a ordem de classificação, a exportação do programa e o menu sair!

Logo após a declaração da função *void*, viemos com a principal que é a *int main*.

#### 4. DESENVOLVIMENTO E RESULTADOS

O trabalho iniciou-se de forma simples e foi se desenvolvendo a cada dia e em grandes proporções. Com o passar do tempo, adquirimos mais ideias e as colocamos em prática de forma que o mesmo foi se desenvolvendo dia após dia. A princípio, era um programa bem rudimentar, porém, conforme fomos adquirindo novas ideias e novos aprendizados, ele foi tomando forma, assim pensamos em como formular a estrutura, o corpo do projeto, imaginando e testando cada tela e menu com suas opções e entradas necessárias. Abaixo, estão os resultados obtidos durante o projeto: Inicialmente, o nosso Menu Principal aparentemente ficou assim:

O código do Menu Principal e menu para a realização dos cadastros dos tempos das equipes apresentado na forma em C.

**Figura 1:** Menu Principal

```
5 void menu(){
6     printf("***** FAST SYSTEM *****\n\n");
7     printf("MENU PRINCIPAL - DIGITE A OPÇÃO DESEJADA: \n\n");
8     printf("1. Bateria 1\n");
9     printf("2. Bateria 2\n");
10    printf("3. Bateria 3\n");
11    printf("4. Obter classificação geral na tela\n");
12    printf("5. Exportar classificação geral\n");
13    printf("6. Sair do sistema\n\n");
14 }
15 void submenuBateria01(int showBat01[5][4]){
16     printf("***** BATERIA 1 *****\n\n");
17     printf("DIGITE O TEMPO DA BATERIA 1 DE CADA UMA DAS EQUIPES: \n\n");
18     printf("1. Tempo da equipe 1 na bateria 1: %02d:%02d:%02d\n", showBat01[0][0], showBat01[0][1], showBat01[0][2]);
19     printf("2. Tempo da equipe 2 na bateria 1: %02d:%02d:%02d\n", showBat01[1][0], showBat01[1][1], showBat01[1][2]);
20     printf("3. Tempo da equipe 3 na bateria 1: %02d:%02d:%02d\n", showBat01[2][0], showBat01[2][1], showBat01[2][2]);
21     printf("4. Tempo da equipe 4 na bateria 1: %02d:%02d:%02d\n", showBat01[3][0], showBat01[3][1], showBat01[3][2]);
22     printf("5. Tempo da equipe 5 na bateria 1: %02d:%02d:%02d\n", showBat01[4][0], showBat01[4][1], showBat01[4][2]);
23     printf("6. Voltar ao menu principal \n\n");
24 }
25
```

**Fonte:** Próprios Autores

O resultado exibido para o usuário, ficou dessa maneira. Abaixo, encontra-se o resultado da tela do Menu Principal apresentando as 3 Baterias, Obtenção da Classificação Geral, a Exportação das Classificações e o Sistema de “SAIDA DO SISTEMA”.

**Figura 2:** Exibição Menu Principal

```
***** FAST SYSTEM *****  
  
MENU PRINCIPAL - DIGITE A OPÇÃO DESEJADA:  
  
1. Bateria 1  
2. Bateria 2  
3. Bateria 3  
4. Obter classificação geral na tela  
5. Exportar classificação geral  
6. Sair do sistema  
  
_
```

**Fonte:** Próprios Autores

As matrizes utilizadas são espaços de memória reservado, utilizado para o armazenamento temporário de informações. Toda vez em que estivermos programando em C, faremos uso das matrizes para colocar as informações na memória que precisamos utilizar.

**Figura 3:** Matrizes

```
212     int bat01[5][4];  
213     int bat02[5][4];  
214     int bat03[5][4];  
215     int melhoresTempos[5][2];  
216     int mediaFinal[5];  
217     int mediaFinalOrdenada[5];  
218     int tempoFinal[5][3];
```

**Fonte:** Próprios Autores

O *switch case* é uma forma de reduzir a complexidade de vários *if* e *else* encadeados. É muito aplicado, principalmente no uso de estruturas de MENU.

**Figura 4:** Switch Case

```

237 while(opcaoPrincipal < 1 || opcaoPrincipal > 6)
238 {
239     scanf("%d", &opcaoPrincipal);
240     switch (opcaoPrincipal)
241     {
242     case 1:
243         system("cls");
244         subopcaoCase01 = 0;
245         while(subopcaoCase01 < 1 || subopcaoCase01 > 6)
246         {
247             system("cls");
248             submenuBateria01(bat01);
249             scanf("%d", &subopcaoCase01);
250
251             switch (subopcaoCase01)
252             {
253             case 1:
254                 cadastroBat01(bat01, subopcaoCase01);
255                 subopcaoCase01 = 0;
256                 break;
257
258             case 2:
259                 cadastroBat01(bat01, subopcaoCase01);
260                 subopcaoCase01 = 0;
261                 break;
262
263             case 3:
264                 cadastroBat01(bat01, subopcaoCase01);
265                 subopcaoCase01 = 0;
266                 break;
267

```

**Fonte:** Próprios Autores

Resultado de como foi apresentado ao usuário o tempo das equipes:

**Figura 5:** Exibição Bateria 1

```

***** BATERIA 1 *****
DIGITE O TEMPO DA BATERIA 1 DE CADA UMA DAS EQUIPES:
1. Tempo da equipe 1 na bateria 1: 01:45:78
2. Tempo da equipe 2 na bateria 1: 01:37:98
3. Tempo da equipe 3 na bateria 1: 01:45:67
4. Tempo da equipe 4 na bateria 1: 02:10:98
5. Tempo da equipe 5 na bateria 1: 02:56:41
6. Voltar ao menu principal
_

```

**Fonte:** Próprios Autores

Em seguida, obtivemos as CLASSIFICAÇÕES DAS EQUIPES.

**Figura 6:** Código que ordena os tempos

```

162 void ordenar(int m_Final[5], int m_FinalOr[5]){
163     m_FinalOr[0] = m_Final[0]; m_FinalOr[1] = m_Final[1]; m_FinalOr[2] = m_Final[2]; m_FinalOr[3] = m_Final[3]; m_FinalOr[4] = m_Final[4];
164     int aux, x, y, i;
165     for(x=0; x<4; x++)
166     {
167         for(y=x+1; y<=4; y++)
168         {
169             if(m_FinalOr[y]<m_FinalOr[x])
170             {
171                 aux=m_FinalOr[y];
172                 m_FinalOr[y]=m_FinalOr[x];
173                 m_FinalOr[x]=aux;
174             }
175         }
176     }
177 }
178

```

Fonte: Próprios Autores

Figura 7: Código que exibe a classificação

```

179 void exibirClassificacao(int mediaDes[5], int mediaOrd[5], int tFinal[5][3]){
180     int x, i;
181     printf("***** CLASSIFICAÇÃO GERAL POR EQUIPES *****\n");
182     for (i=0; i<=4;i++){
183         printf("\n\n %dº Lugar", i+1);
184         for (x=0; x<=4;x++){
185             if(mediaOrd[i] == mediaDes[x]){
186                 printf(" - EQUIPE %d - Tempo (minutos:segundos:mili-segundos): %02d:%02d:%02d", x+1,tFinal[x][0],tFinal[x][1],tFinal[x][2]);
187                 break;
188             }
189         }
190     }
191 }

```

Fonte: Próprios Autores

Figura 8: Exibição da classificação

```

***** CLASSIFICAÇÃO GERAL POR EQUIPES *****

1º Lugar - EQUIPE 2 - Tempo (minutos:segundos:mili-segundos): 01:46:59
2º Lugar - EQUIPE 1 - Tempo (minutos:segundos:mili-segundos): 01:51:57
3º Lugar - EQUIPE 5 - Tempo (minutos:segundos:mili-segundos): 01:55:44
4º Lugar - EQUIPE 3 - Tempo (minutos:segundos:mili-segundos): 02:17:62
5º Lugar - EQUIPE 4 - Tempo (minutos:segundos:mili-segundos): 02:23:73

Digite 1 para voltar ao menu principal _

```

Fonte: Próprios Autores

Logo após a EXPORTAÇÃO:

Figura 9: Código da exportação do arquivo



```

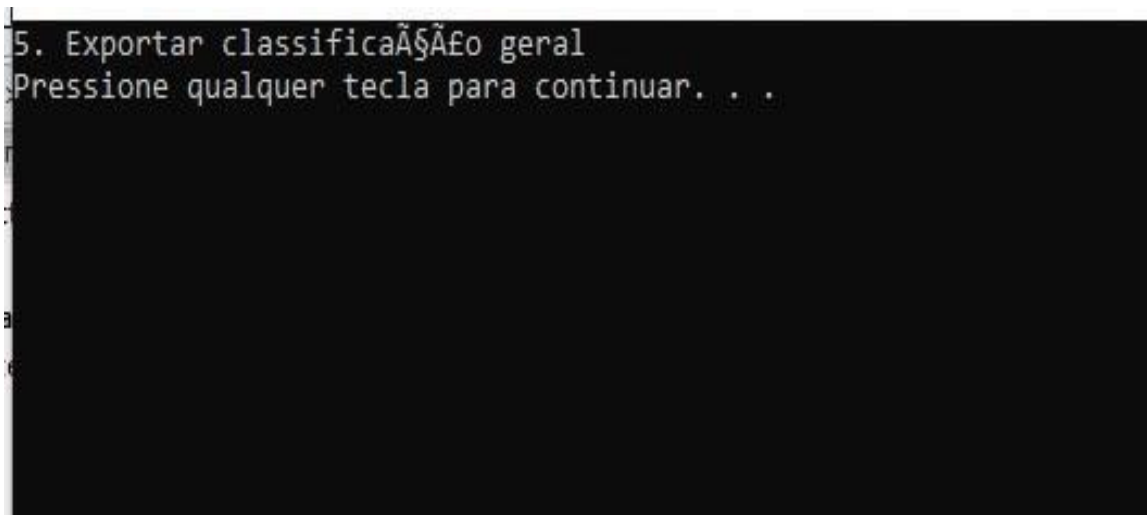
193 void gerarArq(int f_mediaDes[5], int f_mediaOrd[5], int f_tFinal[5][3]){
194     int x, i;
195     FILE *arq;
196     arq = fopen("Resultados.txt", "w");
197     fprintf(arq, "***** CLASSIFICAÇÃO GERAL POR EQUIPES *****\n");
198     for (i=0; i<=4;i++){
199         fprintf(arq, "\n\n %dº Lugar", i+1);
200         for (x=0; x<=4;x++){
201             if(f_mediaOrd[i] == f_mediaDes[x]){
202                 fprintf(arq, " - EQUIPE %d - Tempo (minutos:segundos:mili-segundos): %02d:%02d:%02d", x+1,f_tFinal[x][0],f_tFinal[x][1],f_tFinal[x][2]);
203                 break;
204             }
205         }
206     }
207     fclose(arq);
208 }
209

```

**Fonte:** Próprios Autores

Apresentado ao usuário:

**Figura 10:** Exibição da exportação



**Fonte:** Próprios Autores

**Figura 11:** Exibição da exportação

```

***** CLASSIFICAÇÃO GERAL POR EQUIPES *****

1º Lugar - EQUIPE 2 - Tempo (minutos:segundos:mili-segundos): 01:46:59
2º Lugar - EQUIPE 1 - Tempo (minutos:segundos:mili-segundos): 01:51:57
3º Lugar - EQUIPE 5 - Tempo (minutos:segundos:mili-segundos): 01:55:44
4º Lugar - EQUIPE 3 - Tempo (minutos:segundos:mili-segundos): 02:17:62
5º Lugar - EQUIPE 4 - Tempo (minutos:segundos:mili-segundos): 02:23:73

```

**Fonte:** Próprios Autores

## APERFEIÇOAMENTO DO NOSSO SISTEMA

Após o “término” do Projeto Integrador, notamos que ainda faltava um pouco mais de designer, uma aparência mais chamativa, mais elaborada, e foi então, que decidimos aperfeiçoar o nosso PI e deixá-lo mais

atraente criando um MENU INTERATIVO. Confira abaixo, algumas imagens contendo os códigos utilizados no desenvolvimento do mesmo:

**Figura 12:** Cores do menu

main.c

```
9 void linhaCol(int lin, int col);
10 void box(int lin1, int col1, int lin2, int col2);
11 int menu(int lin1, int col1, int qtd, char lista[3][40]);
12 void textColor(int letras, int fundo);
13 //COR DA LETRA
14 enum{BLACK, //0
15     BLUE, //1
16     GREEN, //2
17     CYAN, //3
18     RED, //4
19     MAGENTA, //5
20     BROWN, //6
21     LIGHTGRAY, //7
22     DARKGRAY, //8
23     LIGHTBLUE, //9
24     LIGHTGREEN, //10
25     LIGHTCYAN, //11
26     LIGHTRED, //12
27     LIGHTMAGENTA, //13
28     YELLOW, //14
29     WHITE //15
30
31 };
32 //COR DO FUNDO
33 enum{ _BLACK=0, //0
34     _BLUE=16, //1
35     _GREEN=32, //2
36     _CYAN=48, //3
37     _RED=64, //4
38     _MAGENTA=80, //5
39     _BROWN=96, //6
40     _LIGHTGRAY=112, //7
41     _DARKGRAY=128, //8
42     _LIGHTBLUE=144, //9
43     _LIGHTGREEN=160, //10
44     _LIGHTCYAN=176, //11
45     _LIGHTRED=192, //12
```

Fonte: Próprios Autores

**Figura 13:** Função que deixa o mouse invisível

```
51
52 void textColor(int letra, int fundo){
53     SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), letra + fundo);
54 }
55
56 void linhaCol(int lin, int col){
57     SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),(COORD){col-1,lin-1}); // coorddenada na tela
58
59     //funcao para deixar o cursor invisivel
60     HANDLE consoleHandle = GetStdHandle(STD_OUTPUT_HANDLE);
61     CONSOLE_CURSOR_INFO info;
62     info.dwSize = 100;
63     info.bVisible = FALSE;
64     SetConsoleCursorInfo(consoleHandle, &info);
65 }
66 void box(int lin1, int col1, int lin2, int col2){
67     int i,j , tamlin, tamcol;
68
69     //achar o tamanho do box
70     tamlin = lin2 - lin1;
71     tamcol = col2 - col1;
72
73     //Monta o Box
74
75     for (i=col1; i<=col2; i++){ // linhas
76         linhaCol(lin1,i);
77         printf("%c",196);
78         linhaCol(lin2,i);
79         printf("%c",196);
80     }
81
82     for (i=lin1; i<=lin2; i++){ //colunas
83         linhaCol(i,col1);
84         printf("%c",179);
85         linhaCol(i,col2);
86         printf("%c",179);
87     }
```

**Fonte:** Próprios Autores

**Figura 14:** Exibição do menu das baterias

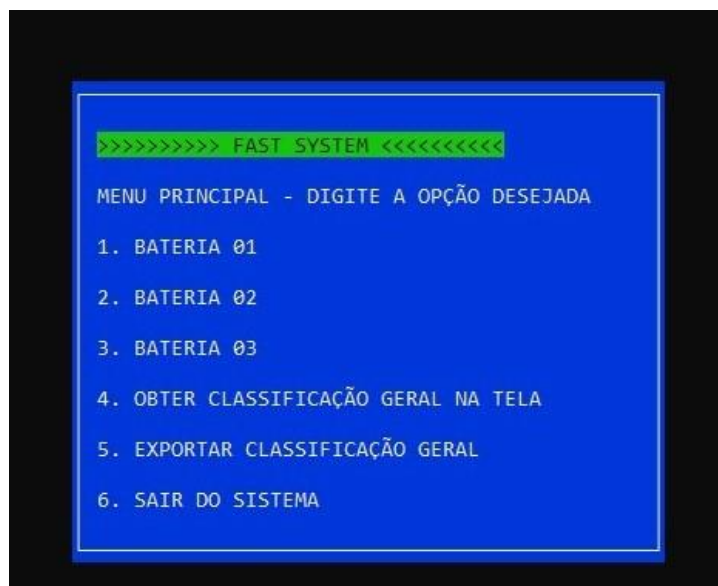
Veja, como ficou o nosso MENU INTERATIVO, após a criação do código:



**Fonte:** Próprios Autores

Posteriormente, com a adição da interação, ficou assim o nosso MENU INTERATIVO. Concluimos que, a nossa ideia ficou muito mais interativa, intuitiva, elegante, atraente e muito mais fácil para o usuário utilizá-lo.

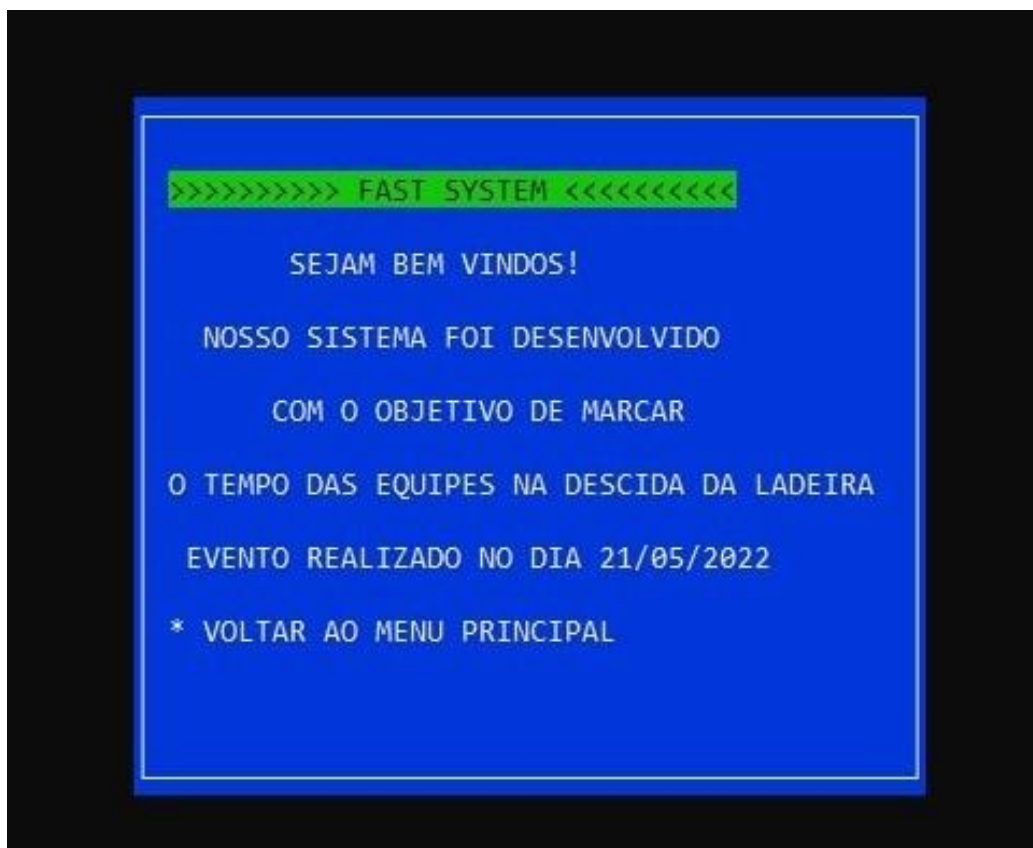
**Figura 15:** Exibição Menu Principal



**Fonte:** Próprios Autores

Ao entrar no primeiro CASE, isto é, no nome da equipe que é Fast System, temos o Menu de Boas-Vindas do sistema, e as suas respectivas opções para que o usuário possa acessar o sistema.

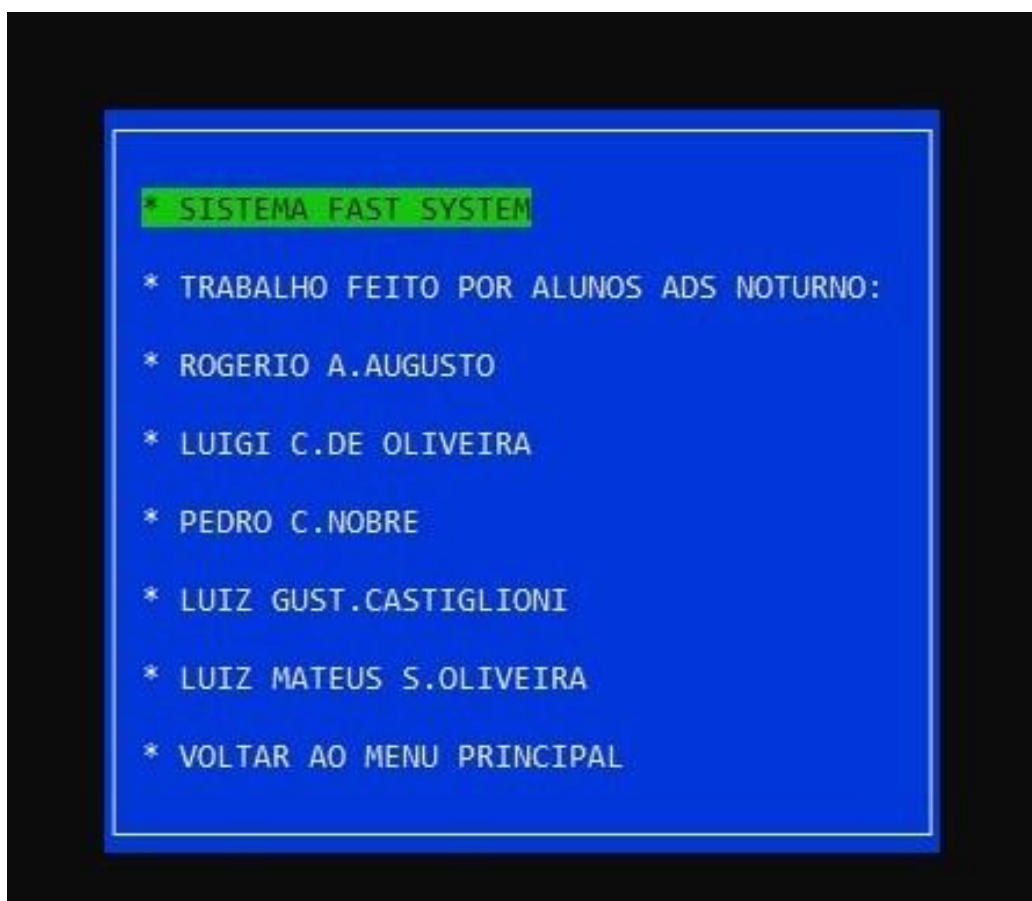
**Figura 16:** Tela de Boas-Vindas



**Fonte:** Próprios Autores

Logo abaixo no menu principal, temos os nomes dos alunos que desenvolveram o sistema (Projeto Integrador).

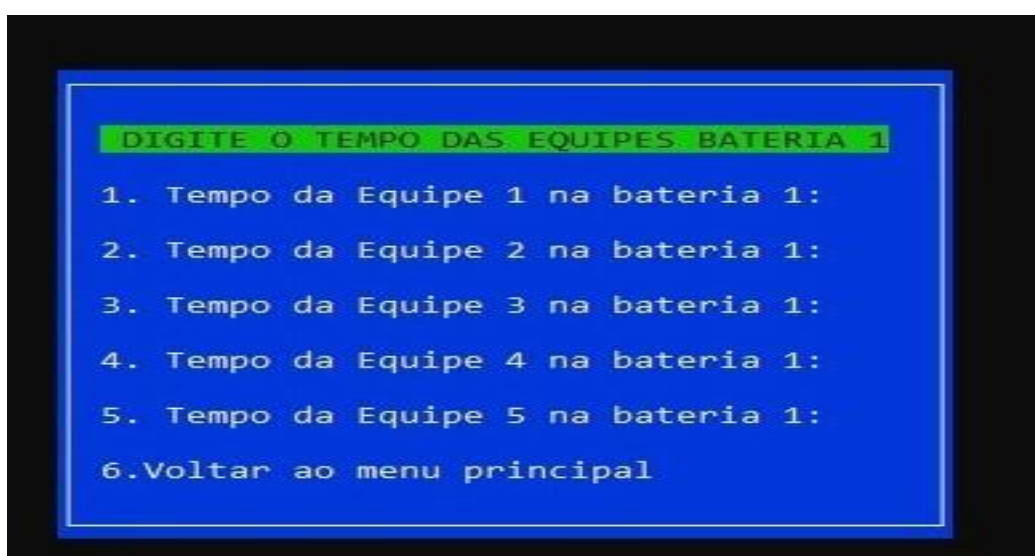
**Figura 17:** Alunos do Projeto



**Fonte:** Próprios Autores

Já na próxima tela, temos uma breve apresentação de como ficou após o aperfeiçoamento na parte das baterias. Para facilitarmos, decidimos deixar sem o MENU INTERATIVO, pois o mesmo deixaria o código maior e achamos por bem, deixá-lo mais objetivo, facilitando o para o usuário (cliente).

**Figura 18:** Menu das Baterias



**Fonte:** Próprios Autores



Em seguida, imprimimos na tela para o usuário, a exportação do sistema com os resultados e suas respectivas classificações.

**Figura 19:** Resultados dos tempos

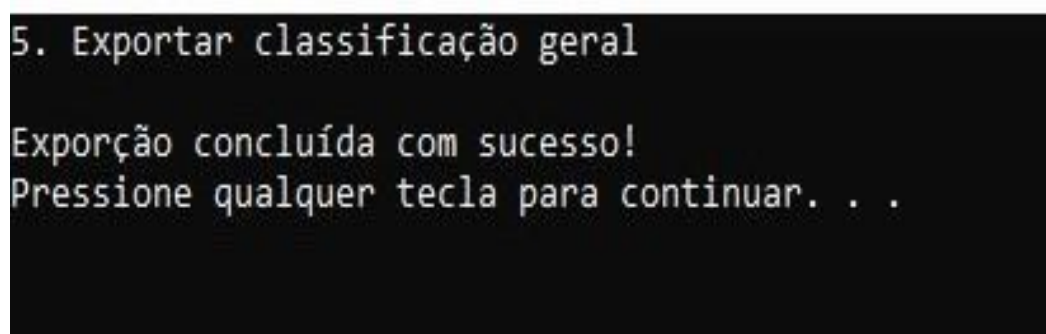


```
Files - EQUIPE 5 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
ce 2º Lugar - EQUIPE 1 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
sources - EQUIPE 2 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
} mair - EQUIPE 3 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
- EQUIPE 5 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
3º Lugar - EQUIPE 1 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
- EQUIPE 2 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
- EQUIPE 3 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
- EQUIPE 5 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
4º Lugar - EQUIPE 1 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
- EQUIPE 2 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
- EQUIPE 3 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
- EQUIPE 5 - Tempo (minutos:segundos:centésimos-segundos): 0:00:00
5º Lugar - EQUIPE 4 - Tempo (minutos:segundos:centésimos-segundos): 0:00:04
Pressione qualquer tecla para continuar. . .
```

**Fonte:** Próprios Autores

Por fim, concluímos o nosso programa apresentando para o usuário, a exportação do sistema.

**Figura 20:** Exibição da Exportação



```
5. Exportar classificação geral
Exporção concluída com sucesso!
Pressione qualquer tecla para continuar. . .
```

**Fonte:** Próprios Autores

Uma mudança significativa que também ajudou na fácil visualização do código foi a incrementação de funções, que economizaram um uso exorbitante de linhas que antes havia no código. A decisão de utilizar



funções veio com o pensamento em utilizar práticas de boa programação, tendo assim um código menor e mais limpo visualmente.

O funcionamento do sistema pode ser visto no vídeo com link ou *qrcode* a seguir:



<https://www.youtube.com/watch?v=Dk8Y2nrJv9Q>

## 5. DISCUSSÃO FINAL

O objetivo do projeto foi o desenvolvimento de um sistema na linguagem de programação C que alocasse os tempos e mostrasse a classificação dos carrinhos de rolimã do evento da descida da ladeira. Procuramos sempre aprimorar o código e suas funções, dando ênfase na fácil legibilidade do mesmo.

Todas as funcionalidades obrigatórias foram desenvolvidas com sucesso. Além disso, foi desenvolvido um outro sistema em que elaboramos um programa diferenciado, onde não utilizaríamos mais o mouse para selecionar as opções, estabelecendo uma interatividade melhor, mais prático e objetivo para o usuário, ganhando cores, destaques e ficando mais visível e intuitivo.

Ao final de tudo, obtivemos um resultado mais do que esperado, pois no começo, não tínhamos muita experiência com Linguagem C. Contudo, durante o processo, foi um pouco difícil e cansativo, porém, uma experiência diferente e legal.

## BIBLIOGRAFIA

DAMAS, Luís, 1951. **Linguagem C**, Luís Damas. Tradução João Araújo Ribeiro. Orlando Bernardo Filho. 10.ed. - Rio de Janeiro: LTC, 2007.

FATECMM. **Descida da Ladeira**. 2022. Disponível em: <https://fatecmm.edu.br>. Acesso em: 18 maio 2022.

RIOMEMORIAS. **Carrinho de Rolimã**. 2020. Disponível em: <https://riomemorias.com.br/memoria/carrinho-de-rolima/>. Acesso em: 18 maio 2022.