

Trabalho Prático / Programação Imperativa

Carlos Brandão e Pedro Simões

2 de janeiro de 2023

Resumo

Este trabalho resume-se num programa que está relacionado com o bem-estar das pessoas, no contexto atual promove a necessidade de realizar atividades físicas com regularidade. Para isso criamos um programa que controla e monitoriza atletas e as suas atividades.

Neste relatório, será possível encontrar passo a passo o código que criámos para o contexto explicado a cima, tendo também presente uma explicação mais aprofundada das funções presentes no programa para uma melhor compreensão do código exibido.

Conteúdo

1	Introdução	2
1.1	Objetivos	2
1.1.1	Objetivo A	2
1.1.2	Objetivo B	2
1.2	Estrutura do documento	2
2	Desenvolvimento	4
3	Dificuldades Encontradas	9
4	Conclusão	10

Capítulo 1

Introdução

Este relatório surge no âmbito do trabalho prático de Programação Imperativa do 1º Ano da Licenciatura de Engenharia de Sistemas Informáticos(Pós-Laboral).

O trabalho prático consiste em aplicar a matéria lecionada para a criação de um programa que visa apoiar, gerir e monitorar a realização de atividades físicas, pretendendo também promover as mesmas.

1.1 Objetivos

1.1.1 Objetivo A

O objetivo A é permitir ao utilizador que seja possível guardar e aceder aos dados sobre atletas, atividades e aos planos das atividades, dados quais estão armazenados em ficheiros.

1.1.2 Objetivo B

O objetivo B é aumentar o conhecimento sobre programação imperativa na linguagem C.

1.2 Estrutura do documento

No Capítulo 1 é introduzido o trabalho juntamente com a apresentação dos objetivos principais.

No Capítulo 2 é onde está presente o desenvolvimento do trabalho prático, neste capítulo vai ter uma explicação aprofundada nos passos que fizemos juntamente com imagens do código para acompanhar a explicação.

No Capítulo 3 é possível encontrar as dificuldades que encontramos durante o nosso trabalho.

No Capítulo 4 é possível encontrar as conclusões que tirámos ao realizar este trabalho.

Capítulo 2

Desenvolvimento

Nós começamos por examinar o enunciado e decidimos começar pela criação das estruturas que iríamos precisar ao longo deste trabalho.

Nós optamos por criar 3 estruturas, uma por cada ficheiro para uma melhor organização de código e dados.

```
typedef struct praticante
{
    char num[5];
    char nome[20];
    int tlm;
    int age;
} Praticante;

typedef struct atividade
{
    char num[5];
    char ativ[15];
    char data[20];
    int duracao, dist;
} Atividade;

typedef struct plano
{
    char num[5];
    char ativ[15];
    int dist;
    char data[20];
    char datain[20];
    char datafim[20];
    char hora[20];
    char horain[20];
    char horafim[20];
} Plano;
```

Figura 1 - Estruturas

De seguida nós optamos por criar um menu onde fosse possível escolher a funcionalidade que quisesse utilizar.

```
do
{
printf("\n-----MENU-----\n");
printf("\n");
printf("1 - Adicionar atleta\n");
printf("2 - Adicionar atividade\n");
printf("3 - Adicionar plano de atividade\n");
printf("4 - Numero atletas por atividade e periodo\n");
printf("5 - Lista de praticantes por atividade e periodo\n");
printf("6 - Mostrar plano de X atleta entre Y e Z datas\n");
printf("7 - Medias de tempos de pratica de atividades dos atletas\n");
printf("8 - Tabela Planos VS Realizados\n");
printf("9 - Tabela Avaliacao\n");
printf("10- Exportar tabela da opcao 9 para ficheiro binario");
printf("11- Sair\n");
printf("\n-----\n");
printf("Opcao: ");
scanf("%d", &op);
printf("\n");
switch (op)
```

Figura 2 - Menu

Assim que concluímos estes passos, começamos a trabalhar nas funcionalidades do nosso programa, e foi assim que criamos a funções para adicionar dados aos três ficheiros.

```
fp = fopen("dados.txt", "a");
if (fp != NULL)
{
fflush(stdin);
printf("Indique o seu codigo (com os zeros, ex: 0123)\n");
fgets(atleta.num, 5, stdin);
atleta.num[5] = '\0'; //a string estava a colocar um /n
fflush(stdin);

printf("Indique o seu nome\n");
fgets(atleta.nome, 20, stdin);
removebarran(atleta.nome); //a string estava a colocar um
fflush(stdin);

printf("Indique o seu numero de telemovel\n");
scanf("%d", &atleta.tlm);
fflush(stdin);

printf("Indique a sua idade\n");
scanf("%d", &atleta.age);
fflush(stdin);

fprintf(fp, "%s;%s;%d;%d\n", atleta.num, atleta.nome, atleta.tlm, atleta.age);
}
fclose(fp);
```

Figura 3 - Função para adicionar atletas (Parte 1)

```
fflush(stdin);
printf("Indique o seu codigo (com os zeros, ex: 0123)\n");
fgets(atleta.num, 5, stdin);
removebarran(atleta.num); // remover /n que fgets coloca desnecessariamente
fflush(stdin);

printf("Indique a data de inicio da atividade (formato: dd-mm-aaaa)\n");
fgets(atleta.datain, 20, stdin);
removebarran(atleta.datain); // remover /n que fgets coloca desnecessariamente
printf("Indique as horas a que comecou (formato: 12h34)\n");
fgets(atleta.horain, 20, stdin);
removebarran(atleta.horain); // remover /n que fgets coloca desnecessariamente
fflush(stdin);
```

Figura 4 - Função para adicionar planos (Parte 1)

```

printf("Indique a data de fim da atividade (formato: dd-mm-aaaa)\n");
fgets(atleta.datafim, 20, stdin);
removebarran(atleta.datafim); // remover /n que fgets coloca desnecessariamente
printf("Indique as horas a que acabou (formato: 12h34)\n");
fgets(atleta.horafim, 20, stdin);
removebarran(atleta.horafim); // remover /n que fgets coloca desnecessariamente
fflush(stdin);

printf("Indique a atividade que planeia realizar (Exemplos: Btt, Natacao, Marcha,\n");
fgets(atleta.ativ, 15, stdin);
removebarran(atleta.ativ); // remover /n que fgets coloca desnecessariamente
fflush(stdin);

printf("Indique quantos kilometros fez (nao pode conter virgulas)\n");
scanf("%d", &atleta.dist);
fflush(stdin);

fprintf(fp, "%s;%s;%s;%s;%s;%s;%d\n", atleta.num, atleta.datain, atleta.horain,
// escreve no ficheiro

```

Figura 5 - Função para adicionar planos (Parte 2)

```

fflush(stdin);
printf("Indique o seu codigo (com os zeros, ex: 0123)\n");
fgets(atleta.num, 5, stdin);
removebarran(atleta.num); // por um motivo desconhecido a string e
fflush(stdin);

printf("Indique a atividade que planeia realizar\n");
fgets(atleta.ativ, 15, stdin);
removebarran(atleta.ativ); // por um motivo desconhecido a string
fflush(stdin);

printf("Indique a data em que planeia realizar a atividade (formato: dd-mm-aaaa)\n");
fgets(atleta.data, 20, stdin);
removebarran(atleta.data);
fflush(stdin);

printf("Indique a duracao da atividade em minutos (horas * 60 = minutos)\n");
scanf("%d", &atleta.duracao);
fflush(stdin);

```

Figura 6 - Função para adicionar atividade (Parte 1)

```

printf("Indique quantos kilometros planeia fazer (nao pode conter virgulas)\n");
scanf("%d", &atleta.dist);
fflush(stdin);

fprintf(fp, "%s;%s;%s;%s;%d\n", atleta.num, atleta.data, atleta.ativ, atleta.duracao);
}

```

Figura 7 - Função para adicionar atividade (Parte 2)

Foi apartir deste momento que demos início às funções do enunciado, começando por ordem criando assim a seguinte função:

Para podermos comparar, as datas nós transformamos as datas num *score*, em que o mês é 100x mais importante que o dia e ano é 2000x mais importante.

```

printf("Indique a atividade (Exemplos: Btt, Natacao, Marcha, etc.)\n");
scanf("%s", atividade);
printf("Indique a data de inicio (formato: dd-mm-aaaa)\n");
scanf("%d-%d-%d", &dia1, &mes1, &ano1);
printf("Indique a data final (formato: dd-mm-aaaa)\n");
scanf("%d-%d-%d", &dia2, &mes2, &ano2);

scoredata1 = dia1 + mes1 * 100 + ano1 * 2000;
scoredata2 = dia2 + mes2 * 100 + ano2 * 2000;

fp = fopen("info_ativ.txt", "r");
if (fp != NULL)
{

```

Figura 8 - Função para pesquisa de atletas (Parte 1)

```

fscanf(fp, "%i;%d-%d-%d;%s;%d\n", &atleta[i].num, &dia[i], &mes[i], &ano[i],
scoredataatleta = dia[i] + mes[i] * 100 + ano[i] * 2000;

if (strcmp(atleta[i].ativ, atividade) == 0 && scoredataatleta > scoredata1 && scoredataatleta < scoredata2)
{
    contador++;
    i++;
}

```

Figura 9 - Função para pesquisa de atletas (Parte 2)

Nós tivemos que trocar o cronograma pois nós tivemos que criar uma função porque por norma a função *fgets* coloca um */n* no final da *string*. A função

removebarran analisa a string vinda de um *fgets* carater por carater , até encontrar o `/n` e substituindo-o por `/0`.

```
void removebarran(char str[])
{
    int i;
    for (i = 0; str[i] != '\0'; i++)
    {
        if (str[i] == '\n')
        {
            str[i] = '\0';
        }
    }
}
```

Figura 10 - Função para remover

De volta ao cronograma, codificamos uma função que apresenta o plano de atividades de um determinado tipo, para um determinado período e praticante.

```

//18
//19
//20
//21
//22
//23
//24
//25
//26
//27
//28
//29
//30
//31
//32
//33
//34
//35
//36
//37
//38
//39
//40
//41
//42
//43
//44
//45
//46
//47
//48
//49
//50
//51
//52
//53
//54
//55
//56
//57
//58
//59
//60
//61
//62
//63
//64
//65
//66
//67
//68
//69
//70
//71
//72
//73
//74
//75
//76
//77
//78
//79
//80
//81
//82
//83
//84
//85
//86
//87
//88
//89
//90
//91
//92
//93
//94
//95
//96
//97
//98
//99
//100
//101
//102
//103
//104
//105
//106
//107
//108
//109
//110
//111
//112
//113
//114
//115
//116
//117
//118
//119
//120
//121
//122
//123
//124
//125
//126
//127
//128
//129
//130
//131
//132
//133
//134
//135
//136
//137
//138
//139
//140
//141
//142
//143
//144
//145
//146
//147
//148
//149
//150
//151
//152
//153
//154
//155
//156
//157
//158
//159
//160
//161
//162
//163
//164
//165
//166
//167
//168
//169
//170
//171
//172
//173
//174
//175
//176
//177
//178
//179
//180
//181
//182
//183
//184
//185
//186
//187
//188
//189
//190
//191
//192
//193
//194
//195
//196
//197
//198
//199
//200
//201
//202
//203
//204
//205
//206
//207
//208
//209
//210
//211
//212
//213
//214
//215
//216
//217
//218
//219
//220
//221
//222
//223
//224
//225
//226
//227
//228
//229
//230
//231
//232
//233
//234
//235
//236
//237
//238
//239
//240
//241
//242
//243
//244
//245
//246
//247
//248
//249
//250
//251
//252
//253
//254
//255
//256
//257
//258
//259
//260
//261
//262
//263
//264
//265
//266
//267
//268
//269
//270
//271
//272
//273
//274
//275
//276
//277
//278
//279
//280
//281
//282
//283
//284
//285
//286
//287
//288
//289
//290
//291
//292
//293
//294
//295
//296
//297
//298
//299
//300
//301
//302
//303
//304
//305
//306
//307
//308
//309
//310
//311
//312
//313
//314
//315
//316
//317
//318
//319
//320
//321
//322
//323
//324
//325
//326
//327
//328
//329
//330
//331
//332
//333
//334
//335
//336
//337
//338
//339
//340
//341
//342
//343
//344
//345
//346
//347
//348
//349
//350
//351
//352
//353
//354
//355
//356
//357
//358
//359
//360
//361
//362
//363
//364
//365
//366
//367
//368
//369
//370
//371
//372
//373
//374
//375
//376
//377
//378
//379
//380
//381
//382
//383
//384
//385
//386
//387
//388
//389
//390
//391
//392
//393
//394
//395
//396
//397
//398
//399
//400
//401
//402
//403
//404
//405
//406
//407
//408
//409
//410
//411
//412
//413
//414
//415
//416
//417
//418
//419
//420
//421
//422
//423
//424
//425
//426
//427
//428
//429
//430
//431
//432
//433
//434
//435
//436
//437
//438
//439
//440
//441
//442
//443
//444
//445
//446
//447
//448
//449
//450
//451
//452
//453
//454
//455
//456
//457
//458
//459
//460
//461
//462
//463
//464
//465
//466
//467
//468
//469
//470
//471
//472
//473
//474
//475
//476
//477
//478
//479
//480
//481
//482
//483
//484
//485
//486
//487
//488
//489
//490
//491
//492
//493
//494
//495
//496
//497
//498
//499
//500
//501
//502
//503
//504
//505
//506
//507
//508
//509
//510
//511
//512
//513
//514
//515
//516
//517
//518
//519
//520
//521
//522
//523
//524
//525
//526
//527
//528
//529
//530
//531
//532
//533
//534
//535
//536
//537
//538
//539
//540
//541
//542
//543
//544
//545
//546
//547
//548
//549
//550
//551
//552
//553
//554
//555
//556
//557
//558
//559
//560
//561
//562
//563
//564
//565
//566
//567
//568
//569
//570
//571
//572
//573
//574
//575
//576
//577
//578
//579
//580
//581
//582
//583
//584
//585
//586
//587
//588
//589
//590
//591
//592
//593
//594
//595
//596
//597
//598
//599
//600
//601
//602
//603
//604
//605
//606
//607
//608
//609
//610
//611
//612
//613
//614
//615
//616
//617
//618
//619
//620
//621
//622
//623
//624
//625
//626
//627
//628
//629
//630
//631
//632
//633
//634
//635
//636
//637
//638
//639
//640
//641
//642
//643
//644
//645
//646
//647
//648
//649
//650
//651
//652
//653
//654
//655
//656
//657
//658
//659
//660
//661
//662
//663
//664
//665
//666
//667
//668
//669
//670
//671
//672
//673
//674
//675
//676
//677
//678
//679
//680
//681
//682
//683
//684
//685
//686
//687
//688
//689
//690
//691
//692
//693
//694
//695
//696
//697
//698
//699
//700
//701
//702
//703
//704
//705
//706
//707
//708
//709
//710
//711
//712
//713
//714
//715
//716
//717
//718
//719
//720
//721
//722
//723
//724
//725
//726
//727
//728
//729
//730
//731
//732
//733
//734
//735
//736
//737
//738
//739
//740
//741
//742
//743
//744
//745
//746
//747
//748
//749
//750
//751
//752
//753
//754
//755
//756
//757
//758
//759
//760
//761
//762
//763
//764
//765
//766
//767
//768
//769
//770
//771
//772
//773
//774
//775
//776
//777
//778
//779
//780
//781
//782
//783
//784
//785
//786
//787
//788
//789
//790
//791
//792
//793
//794
//795
//796
//797
//798
//799
//800
//801
//802
//803
//804
//805
//806
//807
//808
//809
//810
//811
//812
//813
//814
//815
//816
//817
//818
//819
//820
//821
//822
//823
//824
//825
//826
//827
//828
//829
//830
//831
//832
//833
//834
//835
//836
//837
//838
//839
//840
//841
//842
//843
//844
//845
//846
//847
//848
//849
//850
//851
//852

```

Figura 11 - Função para mostrar plano

De seguida, formulámos o código para calcular a média dos tempos dos praticantes que estiveram envolvidos em atividades físicas.

[illegible]

Figura 12 - Função para calcular média

Na fase final do trabalho nós codificamos duas tabelas.

A primeira tabela mostra uma comparação entre os planos do atleta e a atividade que o mesmo realizou.

[illegible]

Figura 13 - Função para a primeira tabela

Para a segunda tabela são pedidas duas datas, uma inicial e uma final, e na tabela são listados os atletas que realizaram uma atividade entre essas datas, mostrando distância percorrida, a duração e a data em que a atividade foi praticada.

```
print(f'Numero | Nome | Data Inicio | Data Fim | Dist Total (km) | Duracao Total (minutos) | Dia em que pratica (dia)')
for i in range(1, n+1):
    scoretotalata[i] = dist[i] + msc[i] * 100 + msc[i] * 2000
    if (scoretotalata[i] > scoretotal MA scoretotalata[i]) : scoretotal = scoretotalata[i]

    print(f'
    print(f'      %s | %s | %s %s %s | %s %s %s | %s | %s | %s %s %s | %s' % (nome[i], data[i],
    )
```

Figura 14 - Função para a segunda tabela

Para concluir o programa, nós codificamos uma função que envia a tabela anterior para um ficheiro binário.

[illegible]

Figura 15 - Função para a exportação da tabela

Capítulo 3

Dificuldades Encontradas

Ao longo do trabalho, deparamo-nos com diversas dificuldades sendo as principais os erros de escrita nos ficheiros, onde eram inseridos /n's não pedidos. Na função *planativatleta*, ao listar a informação no terminal a informação tirada da primeira linha do ficheiro de texto não era lida corretamente. A utilização das datas também foi um desafio, visto que nós tivemos que fazer várias comparações entre as mesmas.

Capítulo 4

Conclusão

Este trabalho prático, colocou-nos vários obstáculos pela frente tendo que os ultrapassar e aumentando assim os nossos conhecimentos. Aprendemos mais sobre trabalhar em equipa e em programar em C.