Relatório TP3

Universidade do Minho

1ºAno MIETI

MP2

Hugo Machado A80362

João Cunha A76645

José Macedo A80311

Introdução

Neste trabalho prático de MP2, foi nos pedido para implementar um jogo de perguntas semelhante ao “Quem Quer Ser Milionário”, com um funcionamento específico e sugerido pelo Professor.

O funcionamento da aplicação consiste em ler as determinadas questões e respostas de um determinado ficheiro de texto (criado a partir de um editor externo), armazenar os dados do ficheiro em memória numa estrutura de dados, mais em concreto usar *listas ligadas generalizadas*.

A nossa aplicação contém um sistema de pontuação, com um *Top 5* de classificados, em vez do utilizador perder mal erre uma questão.

Resolução do problema

Este tema de *listas ligadas generalizadas* foi, em relação com os outros trabalhos práticos o que exigiu maior pesquisa e compreensão de *listas ligadas* e *apontadores,* apos a sua compreensão o seu uso revelou bem pratico mas complicado.

Após a compreensão do tema, a implementação de ideias foi fácil mas encontramos vários erros pelo caminho.

Implementamos duas estruturas da seguinte forma:

```
typedef struct Respostas
{
    char respostaserrada[MAX];
    struct Respostas * next;
}Resp, *AptResp;

typedef struct Pergunta
{
    char pergunta[MAX];
    char respostacerta[MAX];
    struct Respostas * Erradas;
    struct Pergunta * next;
}Perg, *AptPerg;
```

Depois partimos para leitura das questões do ficheiro, que nos deu muito trabalho, até funcionar perfeitamente.

Depois começamos a trabalhar na ideia de atribuir pontuações, e partir desse ponto tivemos que decidir que pontuações iriamos dar (em termos de valores), como iriamos dar os pontos, “fazer registo do nome do utilizador?”, “implementar um top 5 de pontuações?” e também onde guardar as pontuações.

1º Decidimos o seguinte para pontuações:

-Acertar resposta á primeira 11 pontos;

- Acertar resposta á segunda 7 pontos;

- Acertar resposta á terceira 3 pontos;

- Acertar resposta á quarta 0 pontos.

2º O utilizador só passa a fase seguinte se acertar a questão.

3º O utilizador só fica registado caso a sua pontuação seja superior á pontuação do quinto elemento da tabela de pontuações máximas.

4º Os dados da pontuações dos 5 melhores elementos fica guardado num ficheiro a parte para ter um ar mais realista.

Após todo esse trabalho tivemos também de implementar um modo aleatório para distribuir as respostas, encontrar o comando que limpasse o terminal para apresentar a questão e suas repostas de um modo mais limpo e arrumado, e um menu com as pontuações e o início de um novo jogo.

Código

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>
#define MAX 200

typedef struct Respostas
{
  char respostaserrada[MAX];
  struct Respostas * next;
}Resp, *AptResp;

typedef struct Pergunta
{
  char pergunta[MAX];
  char respostacerta[MAX];
  struct Respostas * Erradas;
  struct Pergunta * next;
}Perg, *AptPerg;

AptPerg lerficheiro(AptPerg P)
{
  FILE * Base;
  AptPerg P1 = NULL;
  AptResp R1 = NULL;
  char temp[MAX];

  Base=fopen("Registo.txt","a+");
  fclose(Base);
  Base=fopen("Registo.txt","r");

  if (Base){
         while(fgets(temp,MAX,Base))
         {
                P1=(AptPerg)malloc(sizeof(Perg));
                strcpy(P1->pergunta,temp);
                P1->pergunta[strlen(P1->pergunta)-1]='\0';

                fgets(P1->respostacerta,MAX,Base);
                P1->respostacerta[strlen(P1->respostacerta)-1]='\0';

                R1=(AptResp)malloc(sizeof(Resp));
                fgets(R1->respostaserrada,MAX,Base);
                R1->respostaserrada[strlen(R1->respostaserrada)-1]='\0';
                P1->Erradas=R1;

                R1=(AptResp)malloc(sizeof(Resp));
                fgets(R1->respostaserrada,MAX,Base);
                R1->respostaserrada[strlen(R1->respostaserrada)-1]='\0';
                P1->Erradas->next=R1;

                R1=(AptResp)malloc(sizeof(Resp));
                fgets(R1->respostaserrada,MAX,Base);
                R1->respostaserrada[strlen(R1->respostaserrada)-1]='\0';
                P1->Erradas->next->next=R1;

                P1->next=P;
                P=P1;
         }
   }

   fclose(Base);

   return P;
}

int ordum(AptPerg P)
{
  int pontuacao=0;
  int opc;
  char temp[MAX];
  int i=0;
```

```
  char Pergunt[MAX];
  char Respost[MAX];
  char RespostE1[MAX];
  char RespostE2[MAX];
  char RespostE3[MAX];

  strcpy(Pergunt,P->pergunta);
  strcpy(Respost,P->respostacerta);
  strcpy(RespostE1,P->Erradas->respostaserrada);
  P->Erradas=P->Erradas->next;
  strcpy(RespostE2,P->Erradas->respostaserrada);
  P->Erradas=P->Erradas->next;
  strcpy(RespostE3,P->Erradas->respostaserrada);

  do
  {
          printf("\n->%s\n\n", Pergunt);
          printf("1-%s\n", Respost);
          printf("2-%s\n", RespostE1);
          printf("3-%s\n", RespostE2);
          printf("4-%s\n", RespostE3);

          fgets(temp, MAX, stdin);
          opc=atoi(temp);

          system("clear");

          i=i+1;
          switch(opc)
          {
                  case 1:
                  printf("%d<-Resposta Certa!\n", opc);
                  break;

                  default:
                  printf("%d<-Resposta incorreta!\nTenta outra vez.\n", opc);
          }
  }while(opc!=1);

  if (i==1)
  {
          pontuacao=11;

  }else{
          if (i==2)
          {
                  pontuacao=7;
          }else{
                  if (i==3)
                  {
                          pontuacao=3;
                  }else{
                          pontuacao=0;
                  }
          }
  }
  return pontuacao;
}
int orddois(AptPerg P)
{
  int pontuacao=0;
  char temp[MAX];
  int opc;
  int i=0;

  char Pergunt[MAX];
  char Respost[MAX];
  char RespostE1[MAX];
  char RespostE2[MAX];
  char RespostE3[MAX];

  strcpy(Pergunt,P->pergunta);
  strcpy(Respost,P->respostacerta);
  strcpy(RespostE1,P->Erradas->respostaserrada);
  P->Erradas=P->Erradas->next;
  strcpy(RespostE2,P->Erradas->respostaserrada);
```

```
  P->Erradas=P->Erradas->next;
  strcpy(RespostE3,P->Erradas->respostaserrada);

  do
  {
          printf("->%s\n\n", Pergunt);
          printf("1-%s\n", RespostE1);
          printf("2-%s\n", Respost);
          printf("3-%s\n", RespostE2);
          printf("4-%s\n", RespostE3);

          fgets(temp, MAX, stdin);
          opc=atoi(temp);

          system("clear");

          i=i+1;
          switch(opc)
          {
                  case 2:
                  printf("%d<-Resposta Certa!\n", opc);
                  break;

                  default:
                  printf("%d<-Resposta incorreta!\nTenta outra vez.\n", opc);
          }

  }while(opc!=2);

  if (i==1)
  {
          pontuacao=11;

  }else{
          if (i==2)
          {
                  pontuacao=7;
          }else{
                  if (i==3)
                  {
                          pontuacao=3;
                  }else{
                          pontuacao=0;
                  }
          }
  }
  return pontuacao;
}

int ordtres(AptPerg P)
{
  int pontuacao=0;
  char temp[MAX];
  int opc;
  int i=0;

  char Pergunt[MAX];
  char Respost[MAX];
  char RespostE1[MAX];
  char RespostE2[MAX];
  char RespostE3[MAX];

  strcpy(Pergunt,P->pergunta);
  strcpy(Respost,P->respostacerta);
  strcpy(RespostE1,P->Erradas->respostaserrada);
  P->Erradas=P->Erradas->next;
  strcpy(RespostE2,P->Erradas->respostaserrada);
  P->Erradas=P->Erradas->next;
  strcpy(RespostE3,P->Erradas->respostaserrada);

  do
  {
          printf("->%s\n\n", Pergunt);
          printf("1-%s\n", RespostE1);
          printf("2-%s\n", RespostE2);
          printf("3-%s\n", Respost);
          printf("4-%s\n", RespostE3);
```

```
        fgets(temp, MAX, stdin);
        opc=atoi(temp);

        system("clear");

        i=i+1;
        switch(opc)
        {
                case 3:
                printf("%d<-Resposta Certa!\n", opc);
                break;

                default:
                printf("%d<-Resposta incorreta!\nTenta outra vez.\n", opc);
        }

 }while(opc!=3);

 if (i==1)
 {
        pontuacao=11;

 }else{
        if (i==2)
        {
                pontuacao=7;
        }else{
                if (i==3)
                {
                        pontuacao=3;
                }else{
                        pontuacao=0;
                }
        }
 }
 return pontuacao;
}

int ordquatro(AptPerg P)
{
 int pontuacao=0;
 char temp[MAX];
 int opc;
 int i=0;

 char Pergunt[MAX];
 char Respost[MAX];
 char RespostE1[MAX];
 char RespostE2[MAX];
 char RespostE3[MAX];

 strcpy(Pergunt,P->pergunta);
 strcpy(Respost,P->respostacerta);
 strcpy(RespostE1,P->Erradas->respostaserrada);
 P->Erradas=P->Erradas->next;
 strcpy(RespostE2,P->Erradas->respostaserrada);
 P->Erradas=P->Erradas->next;
 strcpy(RespostE3,P->Erradas->respostaserrada);

 do
 {
        printf("->%s\n\n", Pergunt);
        printf("1-%s\n", RespostE1);
        printf("2-%s\n", RespostE2);
        printf("3-%s\n", RespostE3);
        printf("4-%s\n", Respost);

        fgets(temp, MAX, stdin);
        opc=atoi(temp);

        system("clear");

        i=i+1;
        switch(opc)
        {
                case 4:
```

```
                printf("%d<-Resposta Certa!\n", opc);
                break;

                default:
                printf("%d<-Resposta incorreta!\nTenta outra vez.\n", opc);
        }

  }while(opc!=4);

  if (i==1)
  {
        pontuacao=11;

  }else{
        if (i==2)
        {
                pontuacao=7;
        }else{
                if (i==3)
                {
                        pontuacao=3;
                }else{
                        pontuacao=0;
                }
        }
  }
  return pontuacao;
}

int novojogo(AptPerg P)
{
  char opc;
  int ordem=0;
  int pontuacao=0;
  int soma;

  while(P!=NULL)
  {
        soma=0;
        ordem=(rand()%4)+1;

        switch(ordem)
        {
                case 1:
                soma=ordum(P);
                break;

                case 2:
                soma=orddois(P);
                break;

                case 3:
                soma=ordtres(P);
                break;

                case 4:
                soma=ordquatro(P);
                break;
        }
        getchar();
        system("clear");
        pontuacao=soma+pontuacao;
        P = P->next;
  }

  return pontuacao;
}

void registarpontuacao(int pontuacao)
{
  FILE * Pontos;
  int pontuacoes[5];
  char nomes[5][MAX];
  char nome[MAX];
  char ntemp[MAX];
  char ntemp2[MAX];
  int temp=0;
```

```
  int temp2=0;
  int i;

  Pontos=fopen("Pontuacoes.txt","a+");
  fclose(Pontos);
  Pontos=fopen("Pontuacoes.txt","r");

  for (i = 0; i < 5; i++)
  {
         fgets(nomes[i], MAX, Pontos);
         nomes[i][strlen(nomes[i])-1]='\0';
         fgets(ntemp, MAX, Pontos);
         pontuacoes[i]=atoi(ntemp);
  }

  if (pontuacao>pontuacoes[4])
  {
         printf("\nFicaste no Top 5!!\nQual e o teu nome?");
         fgets(nome, MAX, stdin);
         nome[strlen(nome)-1]='\0';

  }

  if (pontuacao>pontuacoes[0])
  {
         temp=pontuacoes[0];strcpy(ntemp,nomes[0]);
         pontuacoes[0]=pontuacao;strcpy(nomes[0],nome);
         temp2=pontuacoes[1];strcpy(ntemp2,nomes[1]);
         pontuacoes[1]=temp;strcpy(nomes[1],ntemp);
         temp=pontuacoes[2];strcpy(ntemp,nomes[2]);
         pontuacoes[2]=temp2;strcpy(nomes[2],ntemp2);
         temp2=pontuacoes[3];strcpy(ntemp2,nomes[3]);
         pontuacoes[3]=temp;strcpy(nomes[3],ntemp);
         pontuacoes[4]=temp2;strcpy(nomes[4],ntemp2);

  }else{
         if (pontuacao>pontuacoes[1])
         {
                temp=pontuacoes[1];strcpy(ntemp,nomes[1]);
                pontuacoes[1]=pontuacao;strcpy(nomes[1],nome);
                temp2=pontuacoes[2];strcpy(ntemp2,nomes[2]);
                pontuacoes[2]=temp;strcpy(nomes[2],ntemp);
                temp=pontuacoes[3];strcpy(ntemp,nomes[3]);
                pontuacoes[3]=temp2;strcpy(nomes[3],ntemp2);
                pontuacoes[4]=temp;strcpy(nomes[4],ntemp);
         }else{
                if (pontuacao>pontuacoes[2])
                {
                       temp=pontuacoes[2];strcpy(ntemp,nomes[2]);
                       pontuacoes[2]=pontuacao;strcpy(nomes[2],nome);
                       temp2=pontuacoes[3];strcpy(ntemp2,nomes[3]);
                       pontuacoes[3]=temp;strcpy(nomes[3],ntemp);
                       pontuacoes[4]=temp2;strcpy(nomes[4],ntemp2);
                }else{
                       if (pontuacao>pontuacoes[3])
                       {
                              temp=pontuacoes[3];strcpy(ntemp,nomes[3]);
                              pontuacoes[3]=pontuacao;strcpy(nomes[3],nome);
                              pontuacoes[4]=temp;strcpy(nomes[4],ntemp);
                       }else{
                              if (pontuacao>pontuacoes[4])
                              {
                                     pontuacoes[4]=pontuacao;strcpy(nomes[4],nome);
                              }
                       }
                }
         }
  }

  Pontos=fopen("Pontuacoes.txt","w");
  for (i = 0; i < 5; i++)
  {
         fprintf(Pontos, "%s\n%d\n", nomes[i], pontuacoes[i]);
  }
  fclose(Pontos);

}
```

```
void Lerpontuacao()
{
  FILE * Pontos;

  int i;
  char nomes[5][MAX];
  int pontuacoes[5];
  char ntemp[MAX];

  Pontos=fopen("Pontuacoes.txt","r");
  for (i = 0; i < 5; i++)
  {
         fgets(nomes[i], MAX, Pontos);
         nomes[i][strlen(nomes[i])-1]='\0';
         fgets(ntemp, MAX, Pontos);
         pontuacoes[i]=atoi(ntemp);
  }
  fclose(Pontos);

  for (i = 0; i < 5; i++)
  {
         printf("%d-%s\n", pontuacoes[i], nomes[i]);
  }
}


int main()
{
  AptPerg P = NULL;
  char temp[MAX];
  int opc=0;
  int pontuacao=0;
  P = lerficheiro(P);
  srand(time(NULL));

  do{
         system("clear");

         printf("1) Novo Jogo:\n2) Ver pontuacoes:\n0) Sair:\n\n->");

         fgets(temp,MAX,stdin);
         opc=atoi(temp);

         switch(opc)
         {
                case 1:
                       pontuacao=novojogo(P);
                       system("clear");
                       registarpontuacao(pontuacao);
                       break;
                case 2:
                       Lerpontuacao();
                       break;
                case 0:
                       printf("Sair\n");
                       break;

                default:
                printf("Opcao invalida.\n");
         }

  }while(opc!=0);

  return 0;
}
```

Conclusão

Este trabalho prático em conjunto com os outros proporcionou-nos uma ampla abordagem e uma grande capacidade de programação na linguagem C, que nos pode levar a dizer que dominamos minimamente C.