```
#include <stdio.h>
#include <time.h>

/* E-fólio B, 2015/16
   A - carregar um ficheiro de palavras, e indicar quantas linhas tem
       (https://addons.mozilla.org/pt-br/firefox/addon/ortografia-
br/)
   B - mostrar uma palavra aleatória, sem letras acentuadas e todas
minúsculas, indicando número de palavras nestas condições
   C - mostrar uma palavra, sem a parte terminal (única, pelo que é
a única palavra com o prefixo especificado)
   D - mostrar duas palavras, numa está visivel o prefixo, na outra
o sufixo, sendo a parte central comum,
       e sendo também única a parte central
*/

#define MAXSTR 255

// verifica se a palavra está OK, ou seja, apenas tem letras
minúsculas
int PalavraOK(char *str) {
  // primeira letra maiúscula, passar a minúscula
  if(*str>='A' && *str<='Z')
    *str+='a'-'A';
  while(*str!=0 && *str>='a' && *str<='z')
    str++;
  if(*str=='/' || *str=='\n' || *str=='\r')
    *str=0;
  return *str==0;
}

void LibertarPalavras(char **palavras, int nPalavras)
{
  nPalavras--;
  while(nPalavras>0)
    free(palavras[nPalavras--]);
  free(palavras);
}

char **CarregarFicheiro(char *nome, int *nPalavras)
{
  char **palavras;
  char str[255];
  FILE *f;
  int contador=0;

  f=fopen(nome,"rt");

  if(f!=NULL) {
    while(fgets(str,255,f)!=NULL) {
      if(PalavraOK(str))
        contador++;
    }
    fclose(f);

    *nPalavras=contador;

    // repetir carregando as palavras
```

```
    palavras=(char**)malloc(sizeof(char*)*contador);
    if(palavras==NULL) {
      *nPalavras=0;
      return palavras;
    }
    contador=0;

    f=fopen(nome,"rt");
    if(f!=NULL) {
      while(contador>=0 && fgets(str,255,f)!=NULL) {
        if(PalavraOK(str)) {
          palavras[contador]=(char*)malloc(strlen(str)+1);
          if(palavras[contador]==NULL) {
            LibertarPalavras(palavras,contador);
            return NULL;
          }
          strcpy(palavras[contador],str);
          contador++;
        }
      }
      fclose(f);
    }
  } else
    printf("\nFicheiro %s não encontrado.",nome);
  return palavras;
}

// retorna o número de caracteres do prefixo comum
int PrefixoIgual(char *strA, char *strB)
{
  int comum=0;
  while(strA[comum]==strB[comum] && strA[comum]!=0)
    comum++;
  return comum;
}

void ImprimirPalavras(char *palavraA, char*palavraB, int prefixo)
{
  int i;
  for(i=0;i<strlen(palavraA);i++)
    if(i<prefixo)
      printf("%c",palavraA[i]);
    else
      printf("_");
  for(i=strlen(palavraA)-prefixo;i<strlen(palavraB);i++)
    printf("%c",palavraB[i]);
}

/* reutilizado de AFs */
void Baralhar(int v[], int n)
{
  int i, j, aux;
  /* processar todos os elementos */
  for (i = 0; i<n - 1; i++) {
    /* gerar um valor aleatório para sortear o elemento do
    vector a ficar na posição i (entre i e n-1). */
    j = i + rand() % (n - i);
    aux = v[i];
```

```
          v[i] = v[j];
          v[j] = aux;
        }
      }

      /* ver se há uma palavra com um determinado prefixo */
      int EncontraPrefixo(char *prefixo, char**palavras, int contador, int
caracteres)
      {
        int i,j,resultado=-1;
        int *id; // índice para percorrer as palavras por ordem aleatória
        id = (int*)malloc(contador*sizeof(int));
        if (id== NULL)
          return resultado;
        for (i = 0; i < contador; i++)
          id[i] = i;
        Baralhar(id, contador);
        for(i=0;i<contador;i++)
          if(strlen(palavras[id[i]])>strlen(prefixo)+caracteres-1) {
            for (j = 0; prefixo[j] != 0 && palavras[id[i]][j] != 0 &&
palavras[id[i]][j] == prefixo[j]; j++);
            if(prefixo[j]==0) {
              resultado = id[i];
              free(id);
              return resultado;
            }
          }
        free(id);
        return resultado;
      }

      int main (int argc, char **argv)
      {
        FILE *f;
        char str[255];
        int contador=0, iPalavra, prefixo=0, aux, palavraB, i, perguntas=0,
caracteres=1;
        char **palavras=NULL;

        if(argc<4) {
          printf("\nUtilizacao: 111111 <ficheiro de palavras> <perguntas>
<caracteres>");
          return;
        }

        // inicialização do gerador aleatório
        srand(time(NULL));
        rand();

        perguntas=atoi(argv[2]);
        if(perguntas<1)
          perguntas=1;

        caracteres=atoi(argv[3]);
        if(caracteres<1)
          caracteres=1;

        palavras=CarregarFicheiro(argv[1],&contador);
```

```
        printf("carregado %d\n", contador);

        for(i=0;i<perguntas;i++) {
          do {
            do {
              iPalavra=rand()%(contador-2)+1;

              prefixo = PrefixoIgual(palavras[iPalavra], palavras[iPalavra
+ 1]);
              aux = PrefixoIgual(palavras[iPalavra], palavras[iPalavra -
1]);
              if(prefixo>aux)
                prefixo=aux;
            } while (prefixo >= strlen(palavras[iPalavra]) - (caracteres -
1)); // pelo menos X caracteres invisiveis

            palavraB = EncontraPrefixo(palavras[iPalavra] + prefixo,
palavras, contador, caracteres);
          } while (palavraB<0);

          printf("\nPergunta: ");

ImprimirPalavras(palavras[iPalavra],palavras[palavraB],prefixo);
          printf("\tResposta:                                          %s
%s",palavras[iPalavra],palavras[palavraB]);
        }

        LibertarPalavras(palavras,contador);
      }
```