|  |  |  |
| --- | --- | --- |
| **/\*\*Escreva um programa que capture como uma string todo conteúdo de um arquivo de nome**  **escolhido pelo usuário, então converta todas as letras da string para letras minúsculas. Grave**  **o conteúdo modificado de volta no mesmo arquivo. O programa deverá ser capaz de guardar**  **a string obtida, independente de seu tamanho (utilize alocação dinâmica).\*\*/**  **main(){**  **char \*p,nome[100];**  **int c=0,i;**  **FILE \*arq,\*arq2;**  **printf("Digite o nome do arquivo que deseja abrir:");**  **gets(nome);**  **arq = fopen(nome,"r");**  **if(arq==NULL){**  **printf("Arquivo nao encontrado!!!");**  **}else{**  **p=malloc(sizeof(char));**  **while(!feof(arq)){**  **\*(p+c)=fgetc(arq);**  **if(\*(p+c)>=65 && \*(p+c)<=90){**  **\*(p+c)+=32;**  **}**  **c++;**  **p=realloc(p,sizeof(char)\*(c+1));**  **}**  **fclose(arq);**  **arq2=fopen(nome,"w+");**  **for(i=0;i<(c-1);i++){**  **fputc(\*(p+i),arq);**  **}**  **fclose(arq2);**  **}**  **} /// ARQUIVOS >**  \_\_\_\_\_\_\_\_  **vet[i]=97+(rand() % (122-97+1));**  **fputc(vet[i],ponteiro\_arq);**  **/// jogando letras aleatórias no arquivo**  **/// depois verifica ocorrencias**  **printf("Digite um caracter, letras de 'a' ate 'z'\n");**  **letra=getche();**  **while(!feof(arq)){**  **c=fgetc(arq); /// joga os caracteres do arquivo dentro da variavel**  **if(letra==c){**  **cont++;**  **} /// contando ocorrências dentro do arquivo**  **}** | **/\*\* Elabore um programa que gere 350 sequencias de letras aleatórias de tamanho entre 10 e 25 cada.**  **Armazene estas sequencias uma em cada linha de um arquivo de nome escolhido pelo usuário e extensão .txt.**  **Então crie um outro programa que pergunte para o usuário o nome de um arquivo e**  **mostre na tela todas sequencias de caracteres contidas no arquivo que possuem tamanho maior que 20. \*\*/**  **void print\_arquivo(char nome\_arquivo[]);**  **main(){**  **FILE \*ponteiro\_arq;**  **int tam\_sequencia,i,j;**  **char vet[350],nome\_arquivo[20];**  **srand(time(NULL));**  **gets(nome\_arquivo);**  **ponteiro\_arq=fopen(nome\_arquivo,"w");**  **if(nome\_arquivo==NULL){**  **printf("Arquivo nao pode ser aberto! \n");**  **system("pause");**  **}**  **else{**  **for(i=0; i<350; i++){**  **tam\_sequencia=10+(rand()%(25-10+1));**  **for(j=0; j<tam\_sequencia; j++){**  **vet[j]=97+(rand() % (122-97+1));**  **fputc(vet[j],ponteiro\_arq);**  **}**  **fputc('\n',ponteiro\_arq);**  **}**  **}**  **print\_arquivo(nome\_arquivo); /// chamada da função**  **}**  **void print\_arquivo(char nome\_arquivo[]){**  **/// crie uma função que pergunte para o usuário o nome de um arquivo**  **///mostre na tela todas sequencias de caracteres contidas no arquivo que possuem tamanho maior que 20.**  **FILE \*arq;**  **int i,j;**  **char nome[20];**  **char vetor[28];**  **printf("Digite o nome do arquivo: \n");**  **gets(nome);**  **if(strcmp(nome,nome\_arquivo)==0){**  **arq=fopen(nome\_arquivo,"r");**  **for(j=0; j<350; j++){**  **fgets(vetor, 27 ,arq);**  **if(strlen(vetor)>20){**  **printf("%s ",vetor);**  **}**  **}**  **}**  **else{**  **printf("Nome do arquivo está errado! \n");**  **}**  **}** | **Sintaxes ARQUIVOS:**  **R+**: Abre um arquivo texto para leitura e gravação. O arquivo deve existir e pode ser modificado.  **W+**: Cria um arquivo texto para leitura e gravação. Se o arquivo existir, o conteúdo anterior será destruído. Se não existir, será criado.  **A:** Abrir um arquivo texto para gravação. Os dados serão adicionados no fim do arquivo ("append"), se ele já existir, ou um novo arquivo será criado, no caso de arquivo não existente anteriormente.  Rb: Abre um arquivo binário para leitura. Igual ao modo "r" anterior, só que o arquivo é binário.  **Fputs (string, arq);**  **envia uma string para o arquivo.**  **fgets (string, tam\_max, arq);**  **ê uma string do arquivo a string é lida de arq, armazenada em string. O tamanho da string lida é tam\_max, ou até ser encontrado um caractere de fim de linha. O apontador de arq avança de acordo com os caracteres lidos.**  **SPRINTF:**  **sprintf(pfinal, "%s %s %s\n",matriz\_nome[linha\_nome];**  **Estass função concatena strings e formatada elas da forma correta dentro de um vetor, se caso for necessário usa-la, formata no vetor e depois joga pra dentro do arquivo.** |

|  |  |  |
| --- | --- | --- |
| Escreva uma função recursiva que receba um número decimal, e retorne seu equivalente em binário. RECURSIVIDADE  int binario(int n);  main(){  int n;  printf("Digite um valor e vamos converter para binário: \n");  scanf("%i",&n);  int resultado = binario(n);  printf("\nResultado em binario: %i ",resultado);  }  int binario(int n){  if(n==1)  return n;  return (10 \* binario(n/2)) + n%2;  }  /\*\* Escreva uma função recursiva que ajude o usuário com sua dívida.  A função deve receber como parâmetro a quantia ele recebeu de empréstimo,  a taxa de juros mensais e quanto tempo ele irá levar para pagar,  e retornar a quantia que deverá ser paga ao fim do tempo transcorrido.  A solução iterativa se encontra ao lado. (25%) \*\*/  float emprestimo(float v,float j,int t){  if(t==0)  return v;  return emprestimo(v+v\*j,j,t-1);  }  float resultado=emprestimo(valor,juros/100,tempo); /// CHAMADA | /// Escreva uma função recursiva que retorne o maior elemento de um vetor de inteiros.  int procura\_maior(int i,int vet[]);  main(){  int vet[5];  int i;  printf("Preencha o vetor: \n");  for(i=0; i<5; i++){  scanf("%i",&vet[i]);  }  int resultado = procura\_maior(i,vet);  printf("\nO maior numero e %i ",resultado);  }  int procura\_maior(int i,int vet[]){  if(i==1)  return vet[0];  else{  int num = procura\_maior(i-1,vet);  if(vet[i-1]>num){  return vet[i-1];  }  else{  return num;  }  }  } | /\* Escreva uma função recursiva que inverta uma string passada por parâmetro. \*/  #define TAMANHO 20  void invert\_sub(char str[], int i, int j){  if(i>j){  return;  }  int t=str[i];  str[i]=str[j];  str[j]=t;  invert\_sub(str,i+1,j-1);  }  void invert(char str[]){  invert\_sub(str,0,strlen(str)-1);  }  Main(){  invert(str); /// chamada da função  }  /\* Escreva um programa que através da chamada de uma função recursiva retorne a  multiplicação dos dois números passados como parâmetro para a função. Esta função deve  fazer a multiplicação através de somas sucessivas. Ex: 5x3 = 5+5+5. \*/  int soma\_suc(int x, int z){  if(z==1)  return x;  return x+soma\_suc( x, z-1);  }  main(){  printf("%i", soma\_suc(5,3));  } |

STRUCTS

|  |  |  |
| --- | --- | --- |
| void carrega\_arquivo(){ /// CARREGA ARQUIVO  FILE \*arq; /// // logica para iniciar o programa caso o arquivo externo ainda não tenha sido criado  arq=fopen("boteco.dat","rb");  if(arq==NULL){  arq=fopen("boteco.dat","wb");  fclose(arq);  arq=fopen("boteco.dat","rb");  fclose(arq);  }  else{  while(!feof(arq)){  fread(compras\_realizadas+t,sizeof(struct cliente),1,arq);  }  t--;  }  }  void consultar(){ /// CONSULTA  char busca\_nome[50];  int i;  printf("Digite o nome da pessoa que voce deseja consultar os dados: \n");  gets(busca\_nome);  for(i=0; i<t; i++){  if(strstr(cadastro[i].nome,busca\_nome)!=NULL){  printf("%s mora no endereco %s e nasceu %i/%i/%i \n",cadastro[i].nome,cadastro[i].endereco,  cadastro[i].dia,cadastro[i].mes,cadastro[i].ano);  }  }  system("pause");  }  void salva\_arquivo(){ /// SALVA ARQUIVO  FILE \*ponteiro\_arq = fopen("exe3","wb");  fwrite(cadastro, sizeof(struct pessoa), t, ponteiro\_arq);  fclose(ponteiro\_arq);  }  ------------------------------  Crie um programa que auxilie na tarefa de procurar pares para cruza de cães de raça. O  programa deverá possuir um menu. Caso a primeira opção seja escolhida, o usuário deverá  cadastrar um novo cão fornecendo informações como nome, raça, sexo, e idade. Caso a  segunda opção seja escolhida, o usuário deverá informar o nome de um cão, e caso este  esteja cadastrado, o programa deverá mostrar os dados de todos os outros cães do sexo  oposto e da mesma raça do cão informado. Caso a terceira opção seja escolhida, o  programa deverá ser encerrado. As informações cadastradas deverão ser automaticamente  gravadas e carregadas de um arquivo externo. Utilize uma struct para armazenar o cadastro dos cães  \*/  struct cao{  char nome[100];  char sexo;  char raca[100];  int idade;  };  int t=1;  void cadastra(){  struct cao dados[100];  FILE \*arq;  arq=fopen("caes.dat","wb"); | printf("\nDigite o nome do cao:");  gets(dados[t].nome);  fflush(stdin);  printf("\nDigite o sexo do cao:(M ou F)");  dados[t].sexo=getche();  fflush(stdin);  printf("\nDigite a raca do cao:");  gets(dados[t].raca);  fflush(stdin);  printf("\nDigite a idade do cao:");  scanf("%i",&dados[t].idade);  fflush(stdin);  t++;  fwrite(dados,sizeof(struct cao),t,arq);  fclose(arq);  }  void carrega(char nome[100]){  FILE \*arq;  struct cao dados[100];  struct cao caes[100];  char raca[100],sexo;  int i;  arq = fopen("caes.dat","rb");  fread(dados,sizeof(struct cao),100,arq);  for(i=1;i<=t;i++){  if(strcmp(dados[i].nome,nome)==0){  strcpy(raca,dados[i].raca);  sexo=dados[i].sexo;  }  }  printf("\n\n Nome:%s",nome);  printf("\n Raca:%s",raca);  printf("\n Sexo:%c",sexo);  printf("\n\n\n Pares para cruza:\n");  for(i=1;i<=t;i++){  if(strcmp(dados[i].raca,raca)==0 && dados[i].sexo!=sexo){  printf("\nNome: %s",dados[i].nome);  printf("\nRaca: %s",dados[i].raca);  printf("\nIdade: %i",dados[i].idade);  printf("\nSexo: %c\n",dados[i].sexo);  }  }  fclose(arq);  system("pause");  } | int main(){  char r,nome[100];  do{  system("cls");  printf("Digite a acao que deseja executar:\n");  printf("1-Cadastrar cachorro.\n");  printf("2-Consultar pares para cruza.\n");  printf("3-Sair.\n");  r=getche();  switch(r){  case '1':  system("cls");  cadastra();  break;  case '2':  system("cls");  printf("\n\nDigite o Nome de cao que deseja consultar:");  gets(nome);  carrega(nome);  break;  case '3':  printf("\nTenha um Bom dia :)");  break;  default:  printf("\nAcao Invalida!!!");  break;  }  }while(r!='3');  } |