```c
#define ex6
#include "stdio.h"

#ifdef ex1
/*1 - Escreva um programa que tem uma estrutura da dados com os membros abaixo.
    A estrutura é uma variavel local na função main().Receba via teclado o
    conteudo de cada um dos membros numa função e imprima-os no video no
    seguinte fomato(também numa função).

            10        20        30        40        50        60        70
    12345678901234567890123456789012345678901234567890123456789012345678901234567890
        char      int       long                float               double
              unsigned char       unsigned int        unsigned long
  */
 struct variaveis{
    char c;
    int i;
    long l;
    float f;
    double d;
    unsigned char uc;
    unsigned int ui;
    unsigned long ul;
  };

void recebeStruct(struct variaveis *pvar);

  main(){

  struct variaveis var;
  struct variaveis *pvar;
  pvar = &var;
  recebeStruct(pvar);
  imprimeStruct(pvar);
}

//funcao para receber a struct
void recebeStruct(struct variaveis *pvar){
printf("Digite um char: ");
scanf("%c", &pvar->c);
fflush(stdin);
printf("Digite um int: ");
scanf("%d", &pvar->i);
fflush(stdin);
printf("Digite um long: ");
scanf("%ld", &pvar->l);
fflush(stdin);
printf("Digite um float: ");
scanf("%f", &pvar->f);
fflush(stdin);
printf("Digite um double: ");
scanf("%ld", &pvar->d);
fflush(stdin);
printf("Digite um unsigned char: ");
scanf("%c", &pvar->uc);
fflush(stdin);
printf("Digite um unsigned");
scanf("%u", &pvar->ui);
fflush(stdin);
printf("Digite um unsigned long: ");
scanf("%lu", &pvar->ul);
fflush(stdin);
}

//funcao para imprimir a struct
void imprimeStruct(struct variaveis *pvar){
printf("        10        20        30        40        50        60        70\n");
printf("12345678901234567890123456789012345678901234567890123456789012345678901234567890\n");
printf("    %c", pvar->c);
printf("         %-6d", pvar->i);
printf("    %-11ld", pvar->l);
printf("         %-8.1e", pvar->f);
printf("            %-9.1e\n", pvar->d);
printf("          %u", pvar->uc);
printf("                  %u", pvar->ui);
printf("                   %lu", pvar->ul);
}
#endif // ex1


#ifdef ex2
/*
2 - Escreva um programa que receba n valores via teclado, receba também a
    operação a ser executada. Quando for digitado "=" o programa deve mostrar
```

```
85      o resultado acumulado dos n valores. As operações aritmeticas e a entrada
86      de dados devem ser funções que recebe os valores usando ponteiros
87      */
88
89  int somar(int *pn1, int *pn2);
90  int subtrair(int *pn1, int *pn2);
91  int multiplicar(int *pn1, int *pn2);
92  float dividir(int *pn1, int *pn2);
93
94  int main()
95
96  {
97  int entradaDados(char *pc, int *pn1, int *pn2);
98    int n1, n2, *pn1, *pn2, result = 0;
99      char operacao, *pc;
100
101         pn1 = &n1;
102         pc = &operacao;
103         pn2 = &n2;
104
105     result = entradaDados(pc, pn1, pn2);
106     printf("\nResultado: %d\n", result);
107 }
108
109 //funcao entrada
110 int entradaDados(char *pc, int *pn1, int *pn2){
111       int result = 0;
112           printf("Digite um numero e enter, depois digite a operacao e enter: ");
113         scanf("%d", pn1);
114         fflush(stdin);
115     do {
116             gets(pc);
117             if(*pc == '=') {
118                 break;
119             }
120             printf("\nDigite outro numero: ");
121             scanf("%d", pn2);
122             fflush(stdin);
123         switch(*pc) {
124             case '+':  result = somar(pn1, pn2);
125             break;
126             case '-':   result = subtrair(pn1,pn2);
127             break;
128             case '*':   result = multiplicar(pn1,pn2);
129             break;
130             case '/':  result = (int) dividir(pn1,pn2);
131             break;
132         }
133             *pn1 = result;
134
135     }while(*pc != '=');
136 return result;
137 }
138
139 //funcao somar
140 int somar(int *pn1, int *pn2){
141     return *pn1 + *pn2;
142 }
143 //funcao subtrair
144 int subtrair(int *pn1, int *pn2)
145 {
146     return *pn1 - *pn2;
147 }
148 //funcao multiplicar
149 int multiplicar(int *pn1, int *pn2)
150 {
151     return *pn1 * *pn2;
152 }
153 //funcao dividir
154 float dividir(int *pn1, int *pn2)
155 {
156     return (float) *pn1 / (float) *pn2;
157 }
158 #endif // ex2
159
160
161 #ifdef ex3
162 /*3 - Escreva um programa que receba uma letra via teclado. Escreva uma funcao que
163     pesquise esta letra dentro do vetor abaixo. Imprima o resultado da pesquisa no
164     video na funcao main(). O vetor é uma variavel local na função main().Passe
165     como parametro para a funcao o vetor e a letra digitada usando ponteiros.
166     (utilize o comando return)
167     vetor -> b,d,f,h,j,k,m,o,q,s,u,w,y
168 */
```

```c
void verifica(char *pvet, char *pc);

main(){
int i=0;
int ver = 0;
char vet [] = "bdfhjkmoqsuwy";
char c, *pvet, *pc;

printf("Digite uma letra: ");
scanf("%c", &c);

pc = &c;
pvet = vet;
verifica(pvet, pc);

}

//funcao para verificar se existe o char digitado no vetor
void verifica(char *pvet, char *pc){
    int i = 0;
    int ver = 0;
    for(i=0; pvet[i] != '\0'; i++){
           if(pvet[i] == *pc){
             ver ++;
           }
    }
    if(ver > 0){
        printf("A letra digitada consta no vetor");
    }else{
        printf("A letra digitada nao consta no vetor");
    }
}
#endif // ex3

#ifdef ex4
/*
4 - Escreva um programa que receba em 2 funcao 2 strings de ate' 10 caracteres.
    Os vetores sao declaradas como variavel local na função main().
    Escreva uma funcao que recebe as strings com parametros usando ponteiros
    e compare estas 2 strings.
    Retorne como resultado da comparacao 0 se forem DIFERENTES, 1 se forem
    IGUAIS, 2 se a string 1 for maior que a string 2, 3 se a string 2 for maior
    que a string 1 e 4 se as string tem tamanhos iguais mas são diferentes.
*/

int validaString(char *ps1, char *ps2);
char recebeStr1(char *p1);
char recebeStr2(char *p2);

int main()
{
char str1[10], *p1;
char str2[10], *p2;
int result=0;
int fim=0;

p1 = str1;
p2 = str2;

recebeStr1(p1);

recebeStr2(p2);

result = validaString(p1 , p2);

printf("\n Retorna 0 se forem diferentes\n Retorna 1 se forem iguais\n Retorna 2 se a 1
for maior que a 2\n Retorna 3 se a 2 for maior que a 1\n\n");
printf("O resultado e %d\n\n", result);
printf("Digite 1 para finalizar ou qualquer tecla para continuar: ");
}

//funcao para receber a primeira string
char recebeStr1(char *p1){
    printf("Digite uma string: ");
    gets(p1);
     fflush(stdin);
}

//funcao para receber a segunda string
char recebeStr2(char *p2){
 printf("Digite uma string: ");
 gets(p2);
  fflush(stdin);
}
```

```
252
253 //funcao valida string
254 int validaString(char *ps1, char *ps2){
255 int i;
256 int ver=0;
257 int ver1=0;
258 int ver2=0;
259 int ver3=0;
260 int ver4=0;
261
262 for(i = 0; ps1[i] != '\0'; i++){
263     if(ps1[i] != ps2[i]){
264         ver++;
265        break;
266     }
267 }
268 if(ps1[i] == '\0' && ps2[i] == '\0'){
269        printf("iguais");
270        return 1;
271 }else if(ps1[i] == '\0' && ps2[i] != '\0'){
272         return 3;
273         printf("diferentes");
274     }else if(ps1[i] != '\0' && ps2[i] == '\0'){
275         return 2;
276     }
277 }
278 #endif // ex4
279
280 #ifdef ex5
281 /*
282 5 - Escreva um programa com a estrutura abaixo. Defina um vetor de estruturas
283     de 4 elementos.Receba os 4 registros sequencialmente pelo teclado numa
284     função e imprima todos os registros no video em outra função. Faça um menu.
285     Coloque no menu a opção de sair também. Utilize o comando switch.
286     (vetor de estruturas)
287                 nome, end, cidade, estado, cep
288 */
289 struct pessoas{
290  char nome[50];
291  char end[50];
292  char cidade[50];
293  char estado[3];
294  char cep[10];
295 };
296 void recebeStruct(struct pessoas *ps);
297 void imprimeStruct(struct pessoas *ps);
298
299 main(){
300 struct pessoas pessoa[4];
301 struct pessoas *ps;
302 int i=0, fim = 0;
303 ps = pessoa;
304
305 do{
306     recebeStruct(ps);
307     imprimeStruct(ps);
308
309     printf("MENU: \n");
310     printf("1 - SAIR\n2 - CONTINUAR\n");
311
312     scanf("%d", &fim);
313     if(fim == 2){
314         return main();
315     }else if(fim != 1){
316        while(fim != 1 || fim != 2){
317             printf("Comando invalido !\n");a
318             a
319
320             printf("1 - SAIR\n2 - CONTINUAR\n");
321             getchar();
322             scanf("%d", &fim);
323             if(fim == 1 || fim == 2){
324                 break;
325             }
326        }
327     }
328 }while(fim != 1);
329 }
330 //funcao recebe struct
331 void recebeStruct(struct pessoas *ps){
332     int i;
333     for(i = 0; i < 4; i++){
334         fflush(stdin);
335         printf("Digite o %d nome: ", i+1);
```

```
336           gets((ps+i)->nome);
337           fflush(stdin);
338           printf("Digite o %d endereco: ", i+1);
339           gets((ps+i)->end);
340           fflush(stdin);
341           printf("Digite o %d cidade: ", i+1);
342           gets((ps+i)->cidade);
343           fflush(stdin);
344           printf("Digite o %d estado: ", i+1);
345           gets((ps+i)->estado);
346           fflush(stdin);
347           printf("Digite o %d cep: ", i+1);
348           gets((ps+i)->cep);
349           fflush(stdin);
350       }
351   }
352   //funcao imprime struct
353
354   void imprimeStruct(struct pessoas *ps){
355       int i;
356   printf("\nListagem da estrutura:\n\n");
357   for(i = 0; i < 4; i++){
358       printf("nome %d = %s\n",i+1, (ps+i)->nome);
359       printf("endereco %d = %s\n",i+1, (ps+i)->end);
360       printf("cidade %d = %s\n",i+1, (ps+i)->cidade);
361       printf("estado %d = %s\n",i+1, (ps+i)->estado);
362       printf("cep %d = %s\n\n",i+1, (ps+i)->cep);
363   }
364   }
365   #endif // 5
366
367   #ifdef ex6
368   /*
369   6 - Acrescente ao menu do exercicio anterior as funcoes de procura, altera e
370       exclui um registro.
371   */
372   struct pessoas{
373    char nome[50];
374    char end[50];
375    char cidade[50];
376    char estado[3];
377    char cep[10];
378   };
379   void recebeStruct(struct pessoas *ps);
380   void imprimeStruct(struct pessoas *ps);
381   int procuraStruct(struct pessoas *ps);
382   void alteraRegistro(struct pessoas *ps, int j);
383   void excluirRegistro(struct pessoas *ps, int j);
384   main(){
385   struct pessoas pessoa[4];
386   struct pessoas *ps;
387   int i=0, fim = 0, j=0;
388   ps = pessoa;
389
390   do{
391       printf("MENU: \n");
392       printf("1 - IMPRIMIR\n2 - RECEBER DADOS\n3 - BUSCAR\n4 - ALTERAR\n5 - EXCLUIR\n6 -
      SAIR\n\n");
393       scanf("%d", &fim);
394       getchar();
395       if(fim == 1){
396           imprimeStruct(ps);
397
398       }else if(fim == 2){
399           recebeStruct(ps);
400       }else if(fim == 3){
401           j = procuraStruct(ps);
402        if(j == -1){
403           printf("Nenhum registro encontrado\n");
404        }else{
405           printf("nome = %s\n", ps[j].nome);
406           printf("endereco = %s\n", ps[j].end);
407           printf("cidade = %s\n", ps[j].cidade);
408           printf("estado = %s\n", ps[j].estado);
409           printf("cep = %s\n\n", ps[j].cep);
410        }
411
412       }else if(fim == 4){
413           printf("Alterar registro:\n");
414            j = procuraStruct(ps);
415       if(j == -1){
416           printf("Nenhum registro encontrado\n");
417        }else{
418          alteraRegistro(ps, j);
```

```
419       }
420       }else if(fim == 5){
421           printf("Excluir registro:\n");
422            j = procuraStruct(ps);
423       if(j == -1){
424           printf("Nenhum registro encontrado\n");
425        }else{
426          excluirRegistro(ps, j);
427       }
428       }
429   }while(fim != 6);
430   }
431   //funcao recebe struct
432   void recebeStruct(struct pessoas *ps){
433       int i;
434       for(i = 0; i < 4; i++){
435           fflush(stdin);
436           printf("Digite o %d nome: ", i+1);
437           gets((ps+i)->nome);
438           fflush(stdin);
439           printf("Digite o %d endereco: ", i+1);
440           gets((ps+i)->end);
441           fflush(stdin);
442           printf("Digite o %d cidade: ", i+1);
443           gets((ps+i)->cidade);
444           fflush(stdin);
445           printf("Digite o %d estado: ", i+1);
446           gets((ps+i)->estado);
447           fflush(stdin);
448           printf("Digite o %d cep: ", i+1);
449           gets((ps+i)->cep);
450           fflush(stdin);
451       }
452   }
453   //funcao imprime struct
454
455   void imprimeStruct(struct pessoas *ps){
456       int i;
457   printf("\nListagem da estrutura:\n\n");
458   for(i = 0; i < 4; i++){
459       printf("nome %d = %s\n",i+1, (ps+i)->nome);
460       printf("endereco %d = %s\n",i+1, (ps+i)->end);
461       printf("cidade %d = %s\n",i+1, (ps+i)->cidade);
462       printf("estado %d = %s\n",i+1, (ps+i)->estado);
463       printf("cep %d = %s\n\n",i+1, (ps+i)->cep);
464   }
465   }
466   //funcao procura
467   int procuraStruct(struct pessoas *ps2){
468       char nome[100], *ps;
469       ps = nome;
470       int i = 0, ver=0, j=0;
471           fflush(stdin);
472           printf("Digite um nome para buscar:");
473           gets(nome);
474           fflush(stdin);
475   for(j = 0; j < 4; j++){
476           for(i = 0; nome[i] != '\0'; i++){
477               if(ps[i] != ps2[j].nome[i]){
478               break;
479           }
480       }
481       if(ps[i] == '\0' && ps2[j].nome[i] == '\0'){
482          return j;
483       }
484   }
485   return -1;
486   }
487   //funca altera registro
488   void alteraRegistro(struct pessoas *ps, int j){
489    printf("Digite o nome: ");
490               gets(ps[j].nome);
491               fflush(stdin);
492                printf("Digite o endereco: ");
493               gets(ps[j].end);
494               fflush(stdin);
495               printf("Digite a cidade: ");
496               gets(ps[j].cidade);
497               fflush(stdin);
498               printf("Digite o estado: ");
499               gets(ps[j].estado);
500               fflush(stdin);
501               printf("Digite o cep: ");
502               gets(ps[j].cep);
```

```
503             fflush(stdin);
504   }
505
506   void excluirRegistro(struct pessoas *ps, int j){
507             *(ps[j]).nome = '*';
508             *(ps[j]).end = '*';
509             *(ps[j]).cidade = '*';
510             *(ps[j]).estado = '*';
511             *(ps[j]).cep = '*';
512   }
513   #endif // ex6
514
515
516
```