CODE ACADEMY

ОБУЧЕНИЕ ПО ПРОГРАМИРАНЕ

СРЕЩА 15 Динамични структури. Задачи

CODE ACADEMY

Задачи

1. Дадена е квадратна матрица. Напишете функция, която изчислява абсолютната стойност на разликата между сумата по диагоналите.
   Пример
   1 2 3
   4 5 6
   7 8 9
   left-to-right diagonal = 1+ 5 + 9 = 15.
   right to left diagonal = 3 +5 +7 = 17 .

   абсолютната разлика е |15-17| = 2 .

*) Функция, която проверява дали един квадрат е магически

CODE ACADEMY

```
int diagonalDifference(int arr_rows, int arr_columns, int** arr) {

    int l_diagonal_sum = 0, r_diagonal_sum = 0;

    for(int i = 0; i< arr_rows;i++)
    {
        for(int j=0;j<arr_columns;j++)
        {
            if(i == j)
            {
                l_diagonal_sum += arr[i][j];
            }
            if((i+j) == (arr_columns-1))
            {
                r_diagonal_sum += arr[i][j];
            }
        }
    }
    return abs(l_diagonal_sum - r_diagonal_sum);
}
```

CODE ACADEMY

Задачи

2. За даден низ, състоящ се от малки латински букви: Да се дефинира функция, която връща масив от броят на срещанията на всяка малка латинска буква в масива. Функцията има за параметри стринга и дължината на низа. Програмата да прочете от стандартния вход един стринг и да изведе коя буква колко пъти се среща.


```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
int *  numberOfChars(char *s, int n){
     int  *numbers=malloc(26*sizeof(int));
     char c=s[0];
     int i;
     for(i=0 ; c!='\0';i++){
            c=s[i];
            int flag=1;
      for(char letter ='a'; letter<='z' && flag; letter++) {
          if(c==letter){
            int tmp=(int)(letter-'a');
            numbers[tmp]++;
            flag=0;
          }
        }
     }
return numbers;
}
int main (){
    char *duma=(char *) malloc(80*sizeof(char));
    scanf("%s",duma);
    int *broi=(int*)malloc(26*sizeof(int));
    int n=strlen(duma);
    broi=numberOfChars(duma,n);

    for(int k=0;k<26;k++)
        printf("%d\n", broi[k]);
}
```

CODE ACADEMY

Задачи

3. За даден символен низ дефинираме функция, която заменя един символ с друг символ – заместване на всички срещания на символа с друг. Стрингът и двата символа се подават като параметри на функцията.

CODE ACADEMY

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

char *  numberOfChars(char *s, int n, char oldChar, char newChar){
     char  *result=(char *) malloc((n+1)*sizeof(char));
          int i;
          for(i=0; i<n+1;i++){
            if(s[i]==oldChar)
                result[i]=newChar;
            else result[i]=s[i];
          }
return result;
}
int main (){
    char *duma=(char *) malloc(80*sizeof(char));
    char *word=(char *) malloc(80*sizeof(char));
    char c1='a',c2='p';
    printf("string =");
    scanf("%s",duma);
    printf("%s\n", duma);
    int n=strlen(duma);
    word=numberOfChars(duma, n, c1, c2);
    printf("%s", word);

}
```

CODE ACADEMY

Задачи

4. Плъзгащ се прозорец е подмасив с постоянен размер, който се движи отляво надясно през масив. За всяка позиция на прозореца искаме да изчислим някаква информация за елементите в прозореца. Задачата е поддържане на минимума на плъзгащия се прозорец, което означава, че трябва да се намира най-малката стойност във всеки прозорец.
Вход. На стандартния вход за всеки тестов пример се задават две числа – дължината на масива n и дължината на плъзгащия се прозорец m. Следват елементите на масива a1, a2, ..., an.
Ограничения. $1 \leq m \leq n \leq 109$, $1 \leq ai \leq 1000$, $i = 1, 2, ..., n$.
Изход. За всеки тестов пример на стандартния изход на отделен ред се извежда броя на различните най-малки стойности в плъзгащия се прозорец.

CODE ACADEMY

```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

int min(int *subWin, int m){
    int min=*subWin;
    for(int i=1;i<m;i++)
        if(*(subWin+i)<min)
          min=*(subWin+i);

          return min;
}
```

```
int main (){
   int n,m;
   printf("n=");
   scanf("%d",&n);
   printf("m=");
   scanf("%d",&m);
   int *a=(int *)malloc(n*sizeof(int));
   for(int i=0;i<n;i++)
    scanf("%d", a+i);

   for(int w=0;w<n;w+=m){
        int minim=min((a+w),m);
     printf("%d",minim);
   }
```

CODE ACADEMY

Задачи

5. По информационен канал се предават данни във вид на пакети, които трябва да бъдат обработени от вашата програма. Всеки пакет съдържа код и числови данни. Кодът с ключова дума, която определя каква операция трябва да бъде извършена с числовите данни – редица от цели числа.
Кодовете и операциите са:
min – най-малкото число;
max – най-голямото число;
sum – сумата на числата;
num – броя на числата.
От тези 4 кода се образуват още 8, като се добави суфикс p или n.
Суфикс p в кода променя операцията, като от съответната операция се обработват само положителните числа в числовите данни, а суфикс n – само отрицателните.
Вход. На всеки отделен ред от стандартния вход е даден по един пакет: код и данни - редица от цели числа, разделени с интервали.
Ограничения. Числовите данни са цели числа в интервала [−100, 100].
Дължината на редицата, задаваща числовите данни в пакета, е число между 0 и 10.
Изход. За всеки пакет да се изведе на отделен ред на стандартния изход резултата от операцията. Резултатът от операцията върху празна редица е символът *.


```
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
// key=0  all
// key=1  positive
//key=2   negative
int min(int *a,int n, int key){

  int m;
  switch(key){
  case 0:
            m=a[0];
            for(int i=0; i<n;i++)
                if(a[i]<m)
                   m=a[i];
             return m;
  case 1:

            m=1000;
            for(int i=0; i<n;i++)
                if(a[i]>0 && a[i]<m)
                m=a[i];
             return m;
 case 2:    m=1000;
            for(int i=0; i<n;i++)
                if(a[i]<0 && a[i]<m)
                m=a[i];
            return m;
 default: printf("*");
  }
}
```

CODE ACADEMY

```
int max(int *a,int n, int key){
  int max;
  switch(key){
  case 0:   max=a[0];
            for(int i=0; i<n;i++)
                if(a[i]>max)
                   max=a[i];
             return max;
  case 1:  max=1000;
            for(int i=0; i<n;i++)
                if(a[i]>0 && a[i]>max)
                max=a[i];
             return max;
 case 2:   max=0;
            for(int i=0; i<n;i++)
                if(a[i]<0 && a[i]>max)
                max=a[i];
            return max;
 default: printf("*");
  }
}

int sum(int *a,int n, int key){
  int s=0;
  switch(key){
      case 0: for(int i=0; i<n;i++)   s=s+a[i];  return s;
      case 1: for(int i=0; i<n;i++)   if(a[i]>0) s=s+a[i];
return s;
      case 2: for(int i=0; i<n;i++)   if(a[i]<0) s=s+a[i];
return s;
  }
```

CODE ACADEMY

```
int num(int * a, int n, int key){
    int count1,count2;
    switch(key){
    case 0: return n;
    case 1:
        count1 =0;
        for(int i=0;i<n;i++)
            if(a[i]>0) count1++;
        return count1;
    case 2:
        count2 =0;
        for(int i=0;i<n;i++)
            if(a[i]<0) count2++;
        return count2;
    }
}

int (*fun_ptr_arr[])(int *, int, int) = {num, sum, min, max};
```

CODE ACADEMY

```
int main (){
    int key;
    char *kodOp;
    kodOp=(char*)malloc(5*sizeof(char));
    int *data;
    data=(int*)malloc(10*sizeof(int));

    scanf("%s", kodOp);
    int n=strlen(kodOp);
    if(n==4){
        if(kodOp[3]=='p')
            key=1;
        if(kodOp[3]=='n')
            key=2;
    }else{
       if(n==3)
            key=0;
       else printf("Error");
    }
for(int i=0;i<10;i++)
        scanf("%d", &data[i]);
char *kod;
kod=(char*)malloc(3*sizeof(char));
kod[0]=kodOp[0];
kod[1]=kodOp[1];
kod[2]=kodOp[2];
```

CODE ACADEMY

```
int result;
if(strcmp(kod,"num")==0) result=fun_ptr_arr[0](data, 10, key);
 else
  if(strcmp(kod,"sum")==0) result=fun_ptr_arr[1](data, 10, key);
 else
 if(strcmp(kod,"min")==0) result=fun_ptr_arr[2](data, 10, key);
 else
 if(strcmp(kod,"max")==0) result=fun_ptr_arr[3](data, 10, key);
 else
  printf("GRESHEN KOD");
printf("%d", result);
}
```

CODE ACADEMY

Задачи

6. Разглеждаме всички пермутации на три числа

[0,0,0], [0,0,1], [0,0,2], [0,1,0], [0,1,1], [0,1,2], [1,0,0], [1,0,1], [1,0,2], [1,1,0], [1,1,1], [1,1,2]

Извеждаме списък на всички елементи, които нямат зададена сума **n**=3 .

[0,0,0], [0,0,1], [0,0,2], [0,1,0], [0,1,1], [1,0,0], [1,0,1], [1,1,0], [1,1,2]

Вход: четири числа x,y,z – горните граници на числата от всяка позиция n - сумата, която не бива да образуват.
Изход: Функцията отпечатва списък с троиките числа, подреден в лексикографски растящ ред

Използвайте свързан списък или опашка, където да съхранявате подредените тройки
Извикайте функция, която да ви отпечата свързания списък

[ ] [ ]
1 1
1 2
2 1
2 2

CODE ACADEMY

```
for(int i=0 ;i<(x+1);i++)
        for(int j=0 ;j<(y+1);j++)
            for(int k=0 ;k<(z+1);k++)
               if(i+j+k==n)
                   continue
               else:
```

CODE ACADEMY

Задачи

7. Напишете програма, която сортира редица от цели положителни числа в нарастващ ред по броя на единиците в двоичното им представяне. Ако този брой е равен, то числата се сортират в намаляващ ред по тяхната стойност.
Вход. На всеки ред на стандартния вход е зададена редица от не повече от 10000 цели положителни числа, разделени с един или няколко интервала.
Изход. За всеки тест на отделен ред да се изведе редицата от числа, сортирана по искания начин. Числата трябва да са разделени с точно един интервал.

Подзадача 1 `int number1(int n){`

`}`

Подзадача 2 `int * map( int * a, int n){`

CODE ACADEMY

Задачи

8. Да се напише програма, която въвежда ЕГН и проверява дали е валидно ЕГН номер на жени.

Упътване: Проверката за валидност на ЕГН се извършва по следната схема: на всяка цифра от първите девет се съпоставят следните тегла: 2, 4, 8, 5, 10, 9, 7, 3, 6. Всяка от цифрите на ЕГН се умножава по съответното тегло и получените произведения се сумират. Остатъкът при целочислено деление на сумата с 11 е контролно число и трябва да съвпада с десетата цифра от ЕГН.
При остатък 0 или 10 контролната цифра трябва да бъде 0.
В допълнение, тъй като първите 6 цифри представляват рождената дата, те трябва да бъдат валидна дата от годината. Например, ЕГН, започващо с 810229, е невалидно, тъй като 1981 година не е високосна.
Полът се определя от деветата цифра: ако е четно число, то ЕГН е на мъж

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
#include<stdbool.h>

int * toDigit(char *egn){
    int *egnDigits;
    egnDigits=(int*) malloc(10*sizeof(int));
    for(int i=0;i<10;i++)
        egnDigits[i]=egn[i]-'0';

     return egnDigits;
}

int mesec(char * egn){
    return 10*toDigit(egn)[2]+toDigit(egn)[3];
}

int den(char * egn){
    return 10*toDigit(egn)[4]+toDigit(egn)[5];
}

bool isLeapYear(char * egn){
     if((10*toDigit(egn)[0]+toDigit(egn)[1])%4 == 0
        return true;
}
```

```c
bool controlNumber(char * egn){
    bool result=true;
    int weight[]={2, 4, 8, 5, 10, 9, 7, 3, 6};
    int sum=0;
    for(int i=0;i<10;i++)
        sum=sum+toDigit(egn)[i]*weight[i];
    int control=sum%11 ;

    if(control!= toDigit(egn)[9])
         result=false;
     if((control==10) && toDigit(egn)[9]!=0)
         result=true;

    return result;
}

bool isValid(char * egn){
    bool valid=true;

    if (egn[8]%2==0)  valid=false;

    if((mesec(egn)==2) && (den(egn)==29) && !isLeapYear(egn))
            valid=false;

    if(controlNumber(egn)==false)
            valid=false;
 return valid;
}
```

```c
int main(){
    char * s;
    s=(char *)malloc(20*sizeof(char));
    do{
    printf("Enter woman EGN:");
    scanf("%s", s);
    }while(strlen(s)!=10);

    if(isValid(s))
        printf("YES");
    else
        printf("NO");
}
```

CODE ACADEMY

9. Зададен е двоичен стринг, пребройте подстринговете, които започват с 1 и завършват с 1. Например, ако входният стринг е “00100101”, тогава има три подстринга “1001”, “100101” и “101”.Направете задачата, като използвате функция

1010 C_br^2= br*(br-1)/2

CODE ACADEMY

```
9)
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

int count(char * bin, int n){
    int counter=0;
    for(int i=0; i<n;i++){
        counter+=bin[i]-'0';
    }

 return counter*(counter-1)/2;
}

int main(){
    char * s;
    s=(char *)malloc(20*sizeof(char));
    scanf("%s", s);
    printf("%d", count(s,strlen(s)));
}
```

CODE ACADEMY

Задачи за самостоятелна работа

CODE ACADEMY

Задачи за домашна работа

11. **1.** - (преговор списъци) Да се състави програма, чрез която се въвеждат N броя естествени числа от интервала [1..101].
Броят на въведените числа не се знае предварително - определя се от потребителя.
Чрез използване на структура от тип опашка да се изведе въведената редица естествени числа като:
а) първо се извеждат всички нечетни числа;
б) след тях се извеждат всички четни числа;
в) извеждането на числата съответства на реда на въвеждането им.
Пример: 11, 12, 15, 17, 19, 21, 23, 9, 10, 16, 18, 20
Изход: 11, 15, 17, 19, 21, 23, 9, 12, 10, 16, 18, 20

CODE ACADEMY

Задачи за домашна работа

2.- (преговор често използвани функции) Напишете следните функции:

- Функция, която проверявя дали едно число е просто
- Функция, която връща най-големия общ делител на две числа
- Функция, която използва горната функция и по даден като параметър масив от цели числа, връща НОД на елементите от масива

CODE ACADEMY

Задачи за домашна работа

3. Напишете функция, която връща двумерен масив, запълнен по следния начин:

0	20	19	17	14
1	0	18	16	13
2	5	0	15	12
3	6	8	0	11
4	7	9	10	0

CODE ACADEMY

Задачи за домашна работа

4. Напишете функция, която по зададено n връща двумерен масив n x n, елементите на който са подредени като спирала:

Например, при n=5

1	16	15	14	13
2	17	24	23	12
3	18	25	22	11
4	19	20	21	10
5	6	7	8	9

CODE ACADEMY

Задачи за домашна работа

5. Напишете функция, която по даден е масив от цели числа изчислява процента на неговите елементи, които са положителни, които са отрицателни и които са нула. Резултатът, който връща функцията да бъде масив от числа между 0 и 1, даващи пропорцията на всеки вид числа.
Програмата чете масива от числа, извиква горната функция и отпечатва като резултат процентите всеки на нов ред с 6 цифри след десетичната запетая.

Example :

```
arr = [1,1,0, -1,-1]

There are n =5 elements, two positive, two negative and one zero. Their ratios are 2/5=0.400000 , 2/5=0.400000  and 1/5 = 0.200000 . Results are printed as:

0.400000
0.400000
0.200000
```

CODE ACADEMY

Задачи за домашна работа

6. Даден е масив от 5 положителни числа. Да се напишат две функции, които намират **минималната и максималната стойност**, които могат да се изчислят, **като се сумират точно 4 от тези 5 числа.**
Програмата чете масива от тези 5 числа, извиква съответните функции и отпечатва съответно minimum и maximum стойностите, всяка на нов ред.

arr = [1,3,5,7,9]

minimum sum is 1 + 3 + 5 + 7 = 16

maximum sum is 3 + 5 + 7 + 9 = 24.

5. отпечатва се 16 24

CODE ACADEMY

Задачи за домашна работа

7. Дадени са два правоъгълника със страни успоредни на координатните оси. Да се намерят: сумата от лицата им, лицето на обединението и сечението им.
Вход. На всеки от редовете на стандартния вход ще бъде зададена по една двойка правоъгълници с осем цели неотрицателни числа – координати на горния ляв и долния десен ъгъл на единия и другия правоъгълник.
Ограничения. Всички числа са по-малки или равни на 100.
Изход. За всяка двойка правоъгълници, на един ред на стандартния изход програмата трябва да изведе 3 числа — сумата от лицата на двата правоъгълника, лицето на обединението и лицето на сечението им.

Пример:

Вход.	Изход.
0 1 1 0 0 2 2 0	5 4 1
0 1 1 0 2 1 3 0	2 2 0
0 2 3 1 1 3 2 0	6 5 1

CODE ACADEMY

Задачи за домашна работа

8. Времето се задава във 12-часов формат AM или PM. Напишете функция, която по зададено време в такъв формат го конвертира във 24 -часов формат.

9. Note: - 12:00:00AM в 12-часов формат е 00:00:00 в 24-часов формат.
- 12:00:00PM в 12-часов формат е 12:00:00 в 24-часов формат.

Пример

s = '12 : 01: 00PM'

8. Return '12:01:00'.

s = '12 : 01: 00AM'

9. Return '00:01:00'.

CODE ACADEMY

Задачи за домашна работа

9. Една крайна редица от цели числа се нарича зигзаг, ако всеки елемент на редицата (без първия и последния) е или по-голям от двата му съседа или по-малък от двата съседни елемента. Да се напише функция, която по зададен масив от цели числа, определя дали редицата, образувана от тези числа е зигзаг.
Вход. На стандартния вход се задават числови редици – всяка на отделен ред с разделител един интервал между числата.
Ограничения. Всички числа се представят в типа int
Изход. За всяка редица се извежда на отделен ред yes за зигзаг и no – за редица, която не е зигзаг

CODE ACADEMY

Задачи за домашна работа

Задача 10

Да се напише програма, която проверява дали дадена кредитна карта от определен тип— VISA или MasterCard, има валиден номер. За валидиране на номера на кредитните карти, да се използва следното описание:

Номерата на кредитните карти са 16-цифрени в групи по четири.

Например, да проверим дали следната кредитна карта е валидна:

4 2 0 4 – 5 8 7 6 – 9 0 1 2 – 5 2 3 4

1) Обръщаме цифрите на номера и получаваме:

4 3 2 5 2 1 0 9 6 7 8 5 4 0 2 4

2) Удвояваме цифрите, стоящи на четна позиция, и получаваме:

4 6 2 10 2 2 0 18 6 14 8 10 4 0 2 8

3) Ако получим събираеми, по-големи от 9, ги разделяме на две отделни цифри. Събираме така получените цифри:

4 + 6 + 2 + 1 + 0 + 2 + 2 + 0 + 1 + 8 + 6 + 1 + 4 + 8 + 1 + 0 + 4 + 0 + 2 + 8 = 60

4) Ако сборът се дели на 10 без остатък, то кредитната карта е валидна. Ако номерът започва с 4, то типът ѝ е VISA, а ако има префикс 51–55, тя е MasterCard.