

Работа над ошибками контрольной.

Задача 1. Дана матрица $X[0:n-1][0:m-1]$ и массив $Y[0:k-1]$. Написать программу, которая вычисляет массив Z , состоящий из элементов X , расположенных между первым четным и максимальным нечетным элементами каждого столбца, если все эти элементы присутствуют в массиве Y . Ввод данных и вычисления оформить в виде отдельных функций.

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

```
void input_mass(int b[], int *k) //k передаём по адресу
```

```
{int i;  
  puts("Input k"); scanf("%d",k);  
  puts("Input array");  
  for(i=0;i<*k;i++)  
    scanf("%d",&b[i]);  
}
```

```
//для матрицы передаём массив указателей на строки
```

```
void input_matr(int a[][10], int *n, int *m)
```

```
{int i,j;  
  puts("Input n,m"); scanf("%d%d",n,m);  
  puts("Input matrix");  
  for(i=0;i<*n;i++)  
    for(j=0;j<*m;j++)  
      scanf("%d",&a[i][j]);  
}
```

```

void count (int k, int Y[], int n, int m, int X[][10], int Z[], int *nz)
{ int i,j,t,nch,nmax,flag;

*nz=0; //длина нового массива
for(j=0;j<m;j++)
{
nch=nmax=-1; //оба индекса могут быть не найдены
for(i=0;i<n;i++)
{
if (X[i][j]%2==0&& nch==-1) nch=i; //если чётный и первый
if (X[i][j]%2!=0) //если нечётный
if (nmax==-1) nmax=i; //если первый
else
if (X[i][j]>X[nmax][j]) nmax=i; /* если не первый –
сравниваем с максимумом */
}
}
}

```

```

if (nch!=-1&&nmax!=-1) //если оба найдены
{
if (nch>nmax) //упорядочиваем чтобы nmax >= nch
    t=nmax, nmax=nch, nch=t;
if (nmax-nch>1) //между ними есть элементы
    { flag=1; //признак присутствия всех
    for(i=nch+1; i<nmax&&flag; i++)
        { //проверка присутствия элемента X[i][j] в Y[0:k-1]
        for (t=0; t<k&&X[i][j]!=Y[t]; t++);
        if (t==k) flag=0; //если отсутствует
        }
    if (flag) //присутствуют все
//копируем элементы из матрицы в массив
        for(i=nch+1; i<nmax; i++)
            Z[(*nz)++]=X[i][j];
        }
    }
}
}
}

```

```
void output_c(int nc, int c[])
{ int i;
  if (nc==0) puts("No array c");
  else
  {
    puts("array c");
    for(i=0;i<nc;i++)
      printf("%7d",c[i]);
  }
}
```

```
int main()
{
  int a[10][10],n,m,b[10],k,c[50],nc;

  input_matr(a, &n, &m);
  input_mass(b, &k);
  count (k,b,n,m, a, c,&nc);
  output_c(nc,c);
  return 0;
}
```

Задача 2. Дана символьная строка. Заменить каждую подстроку, заключенную в круглые скобки и состоящую из цифр, звездочками. Количество звездочек равно первой цифре в подстроке. Преобразование строки оформить как отдельную функцию.

```
#include <stdio.h>
#include <ctype.h>
#define maxln 81
int zamena(char s[])
{
    char *s1,*s2;
    int i,n,f=0;
    while (*s) //пока не конец строки
    if (*s=='(') //найдено начало подстроки, этот адрес заносим в s1
    {
        s1=++s;
        while (*s&&isdigit(*s))s++; //пропускаем цифры
        if (*s==')'&&s1!=s) //если за цифрами скобка и скобки не рядом
```

```
{  
    f=1; //строка преобразовывалась  
    n=*s1-'0'; //вычисляем количество цифр  
    for(s2=s1;*s;*s2++=*s++); //удаляем подстроку  
    *s2='\0';  
    for(;s1<=s2;s2--) //освобождаем место для вставки '*'  
        *(s2+n)=*s2;  
    for(i=0;i<n;i++) //вставляем звездочки  
        *(s1++)='*';  
    s=s1;  
}  
}  
else  
    s++; //если не скобка, идем к следующему символу  
return f;  
}
```

```
int main ()
{
    char str[maxIn];
    int f;
    puts("input string");
    gets(str);
    f=zamena(str);
    if (f==0)
        puts("No changes");
    else
    {
        puts("Modified string ");
        puts(str);
    }
    return 0;
}
```


Домашнее задание (0.3 балла)

Задачу 1 решают студенты с нечётными номерами вариантов по лабораторным, задачу 2 - с чётными. Сдача д/з на ближайшей лекции.

Задача 1. Дана символьная строка. Написать функцию для выделения подстрок, заключенных в комментарии `/* ... */` и содержащих только латинские буквы. Привести вызов функции. Вывести полученные подстроки на экран.

Задача 2. Дана символьная строка. Написать функцию для выделения подстрок, содержащих только латинские буквы и/или цифры и имеющих чётную длину. Привести вызов функции. Вывести полученные подстроки на экран.

- Ввод данных и вывод результата осуществить в главной функции.
- Выделение подстрок оформить как отдельную функцию с параметрами.

Структуры

Задача 2. Даны структуры вида



Ввести структуры в массив и найти в нем самую дорогую книгу.

Ввод данных и поиск оформить как отдельные функции.

Функция поиска вернет указатель на самую дорогую книгу.

Рассмотрим два варианта ввода

1)

Книга # 1

Автор.....<Автор 1>

Название.....<Название1>

Год издания.....<Год 1>

Цена.....<Цена 1>

Продолжаете ввод? (y/n)

...

2)

Число книг в каталоге =<k>

Книга # 1

Автор..... <Автор 1>

Название..... <Название1>

Год издания..... <Год 1>

Цена..... <Цена 1>

Книга # 2

...

Будем использовать 1-й вариант ввода.

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

```
#define Imax 200
```

```
//пропуск символов до конца строки
```

```
#define CLR while (getchar()!='\n')
```

```
struct book
```

```
{
```

```
char author[20], name[60];
```

```
int year, price;
```

```
};
```

//Функция ввода массива структур (каталога)

```
void readcat(int *kol, book cat[])
{ char ch;
  *kol=0;
  do
  { printf("Book # %d\n", ++(*kol));
    printf("Author....."); gets(cat->author);
    printf("Title....."); gets(cat->name);
    printf("Publishing year.."); scanf("%d", &cat->year);
    printf("Price....."); scanf("%d", &cat->price); CLR;
    printf("Continue ? (y/n) ");
    ch=getchar();
    CLR;
    cat++;  }
  while ((ch=='Y' || ch=='y')&&*kol<=lmax);
}
```

//Функция поиска самой дорогой книги

```
book *dorog(int kol, book cat[])
{
    int i, max=0; book *dorkn;
    for (i=0; i<kol; i++, cat++)
        if (cat->price>max) max=cat->price, dorkn=cat;
    return (dorkn);}

int main()
{
    int kolknig; book catalog[200], *dorkniga;
    readcat(&kolknig, catalog);
    dorkniga=dorog(kolknig, catalog);
    printf("Most expensive book:\n");
    printf("Author.....%s\n", dorkniga->author);
    printf("Title.....%s\n", dorkniga->name);
    printf("Publishing year..%d\n", dorkniga->year);
    printf("Price.....%d\n", dorkniga->price);
    return 0;}
```