



Язык С

Лекция №3

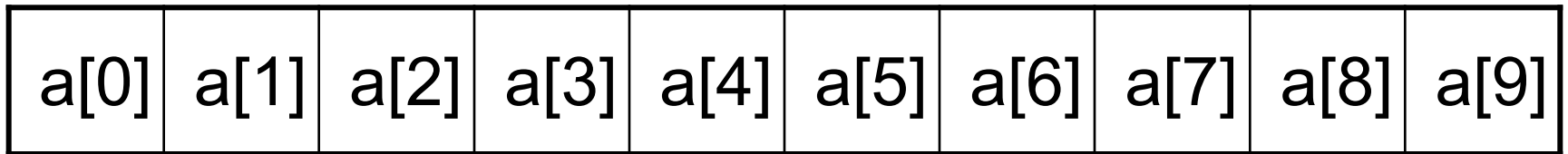
Массивы и указатели



Массивы

Одномерные массивы

```
int a[10];
```



Нумерация индекса начинается с нуля!

В описании указывается число элементов, а не верхняя граница

Выход за пределы индекса – одна из самых распространенных ошибок!

Многомерные массивы

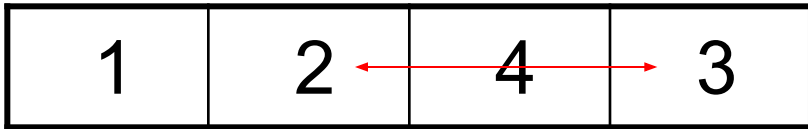
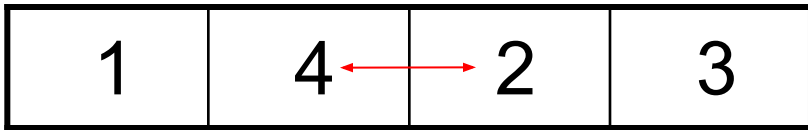
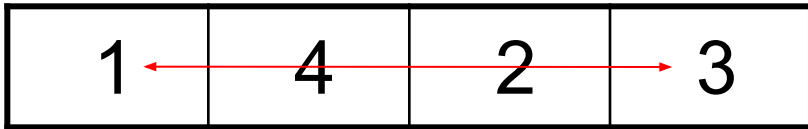
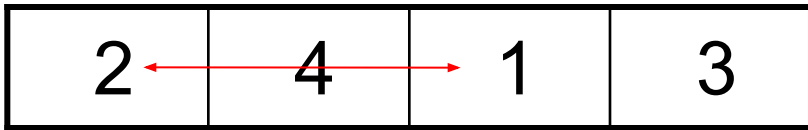
```
int a[20][8];
```

Нумерация всех индексов начинается с нуля!

Классические задачи, возникающие при работе с массивами

- Ввод/вывод
(сохранение в файл, передача по сети и т.п.)
- Сортировка
(упорядочение по какому-либо признаку, не прибегая к дублированию массива)
- Поиск
(найти номер элемента массива по его значению)

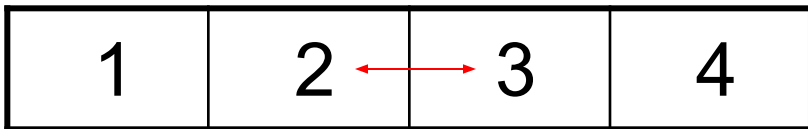
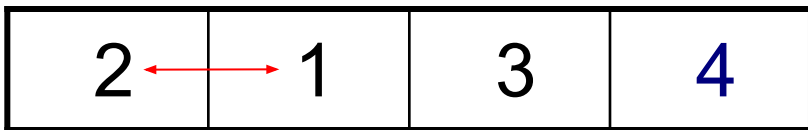
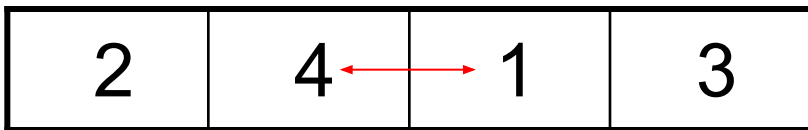
Линейная сортировка



Число просмотров $\sim \frac{1}{2} (N-1)^2$

Число перестановок $\sim \frac{1}{4} (N-1)^2$

Сортировка «пузырьком»



Число просмотров $\sim \frac{1}{2} (N-1)^2$

Число перестановок $\sim \frac{1}{4} (N-1)^2$

Можно остановиться, как только не будет ни одной перестановки

Ввод массива

```
#define N 10
#include <stdio.h>
main()
{
    int a[N], x;
    int i, j;

    /* Ввод массива */
    for (i=0; i<N; i++)
    { printf("a[%d]=", i);
      scanf("%d", &a[i]);
    }
}
```

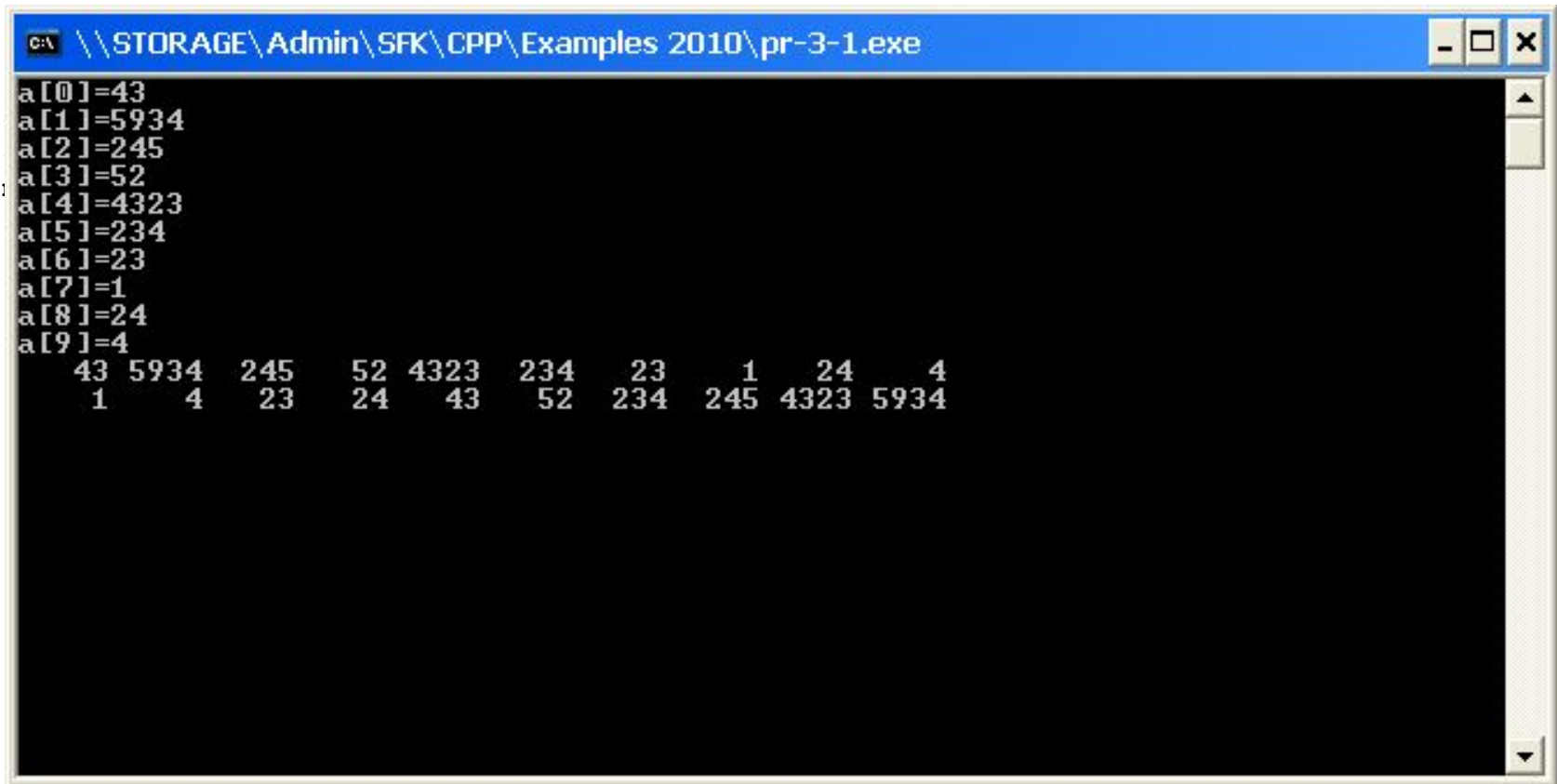

Сортировка массива

```
/* Линейная сортировка массива */  
for (i=0; i<N-1; i++)  
    for (j=i+1; j<N; j++)  
        if (a[i]>a[j])  
            {x=a[j];  
              a[j]=a[i];  
              a[i]=x;  
            }
```

Вывод массива

```
/* Вывод отсортированного массива */  
for (i=0; i<N; i++) printf ("%5d", a[i]);  
printf ("\n");  
  
return 0; // Код возврата  
}
```

Вся программа целиком



```
C:\ \\STORAGE\Admin\SFK\CPP\Examples 2010\pr-3-1.exe
a[0]=43
a[1]=5934
a[2]=245
a[3]=52
a[4]=4323
a[5]=234
a[6]=23
a[7]=1
a[8]=24
a[9]=4
  43 5934  245  52 4323  234  23  1  24  4
    1  4  23  24  43  52  234  245  4323  5934
```

Бинарный поиск

5	14	21	45	55	60	72	73	89	93
0	1	2	3	4	5	6	7	8	9

1. $L=0$; $R=9$; $N=(L+R)/2$; // $N==4$; $A[N]==55$; $A[N]<72$
2. $L=N+1$; $R=9$; $N=(L+R)/2$; // $N==7$; $A[N]==73$; $A[N]>72$
3. $L=5$; $R=N$; $N=(L+R)/2$; // $N==6$; $A[N]==72$;

Бинарный поиск

```
// Бинарный поиск элемента массива равного b  
int L, R, n;  
bool f = true;  
  
L=0; R=N-1;  
do {  
    n=(L+R)/2;  
    if (a[n]==b) { f=false; break; }  
    if (a[n]<b) L=n+1; else R=n;  
}  
while (L<=R);  
if (f) n=-1;  
}
```



Строки СИМВОЛОВ

Строка символов – массив

```
char str[12] = "Borland C++";
```

B	o	r	l	a	n	d		C	+	+	\0
---	---	---	---	---	---	---	--	---	---	---	----

Ввод строки с клавиатуры:

```
scanf("%s",str); или gets(str);
```

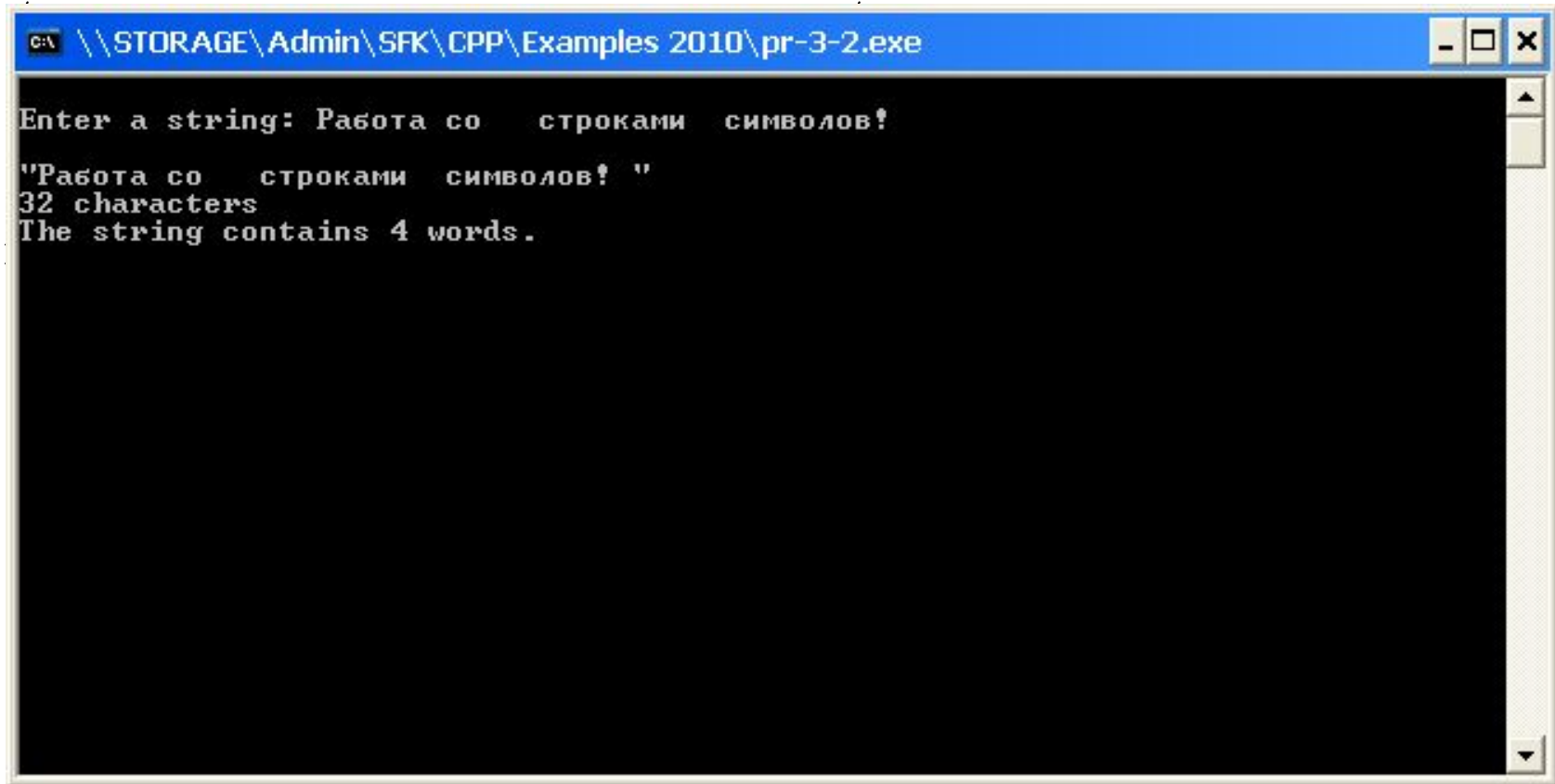
Вывод строки:

```
printf("%s",str); или puts(str);
```

Функции для работы со строками

- **strcpy(s1,s2)** – копирует содержимое строки s2 в s1, возвращает указатель на s1
- **strcat(s1,s2)** – присоединяет строку s2 к s1, возвращает указатель на s1
- **strlen(s1)** – возвращает длину строки, последний нулевой байт не учитывается
- **strcmp(s1,s2)** – сравнивает строки, возвращает положительное, нулевое или отрицательное значение

Простейшая программа работы со строками



```
C:\ \STORAGE\Admin\SFK\CPP\Examples 2010\pr-3-2.exe
Enter a string: Работа со строками символов?
"Работа со строками символов! "
32 characters
The string contains 4 words.
```

```
    if ((s[i]!=' ') && (s[i+1]==' ')) k++;
printf("The string contains %d words.\n",k);
}
```

Копирование строки

```
while (s2[i]=s1[i]) i++;
```

ИЛИ

```
while (s2[i]=s1[i++] );
```



Указатели

Типизированные указатели

`char *c; // указатель на char`

`int *i, j; // указатель на int и просто int`

`i=&j; // i присвоить адрес j`

`*i=1; // разыменованный указатель j=1`

Указатели на void

`void *p;` // нетипизированный указатель

`float *pf, f;` // типизированный указатель

`pf=&f;` // pf присвоить адрес f

`p=pf;` // одному указателю присвоить значение другого

`pf=(float *) p;` // явное указание типа при разыменовании

Указатели и массивы

```
int a[10], * p;
```

```
char *p;
```

```
char str="Strings With Capital Words";
```

```
p=str;
```

```
while (*p) putc(*p++);
```

Указатели и массивы

`p=a;`

`p=&a[0];`

`a[5]`

`*(p+5)`

Массивы указателей

```
char *ext[]={"exe", "com", "dat", "c", "pas", "cpp"}
```

```
int **p;
```

```
printf("%s",ext[0]);
```


Динамическое размещение данных

- Неинициализированный указатель

int *p;

- Выделение памяти (N элементов)

p=malloc(sizeof(int)*N);

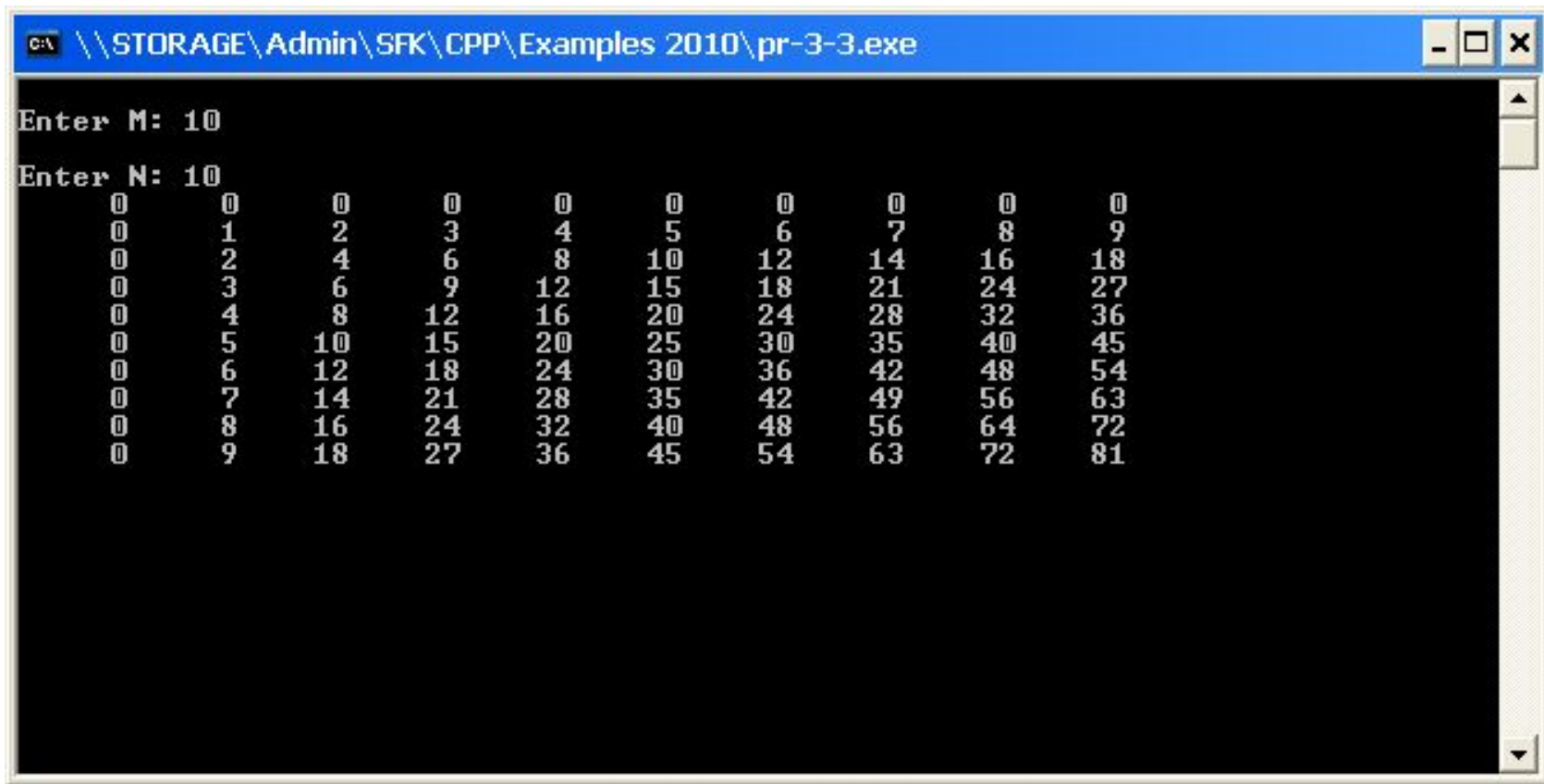
- Использование

p[i] либо *(p+i)

- Освобождение памяти

free(p);

Динамическое создание двумерного массива



```
C:\ \\STORAGE\Admin\SFK\CPP\Examples 2010\pr-3-3.exe
Enter M: 10
Enter N: 10
 0  0  0  0  0  0  0  0  0  0
 0  1  2  3  4  5  6  7  8  9
 0  2  4  6  8 10 12 14 16 18
 0  3  6  9 12 15 18 21 24 27
 0  4  8 12 16 20 24 28 32 36
 0  5 10 15 20 25 30 35 40 45
 0  6 12 18 24 30 36 42 48 54
 0  7 14 21 28 35 42 49 56 63
 0  8 16 24 32 40 48 56 64 72
 0  9 18 27 36 45 54 63 72 81
```

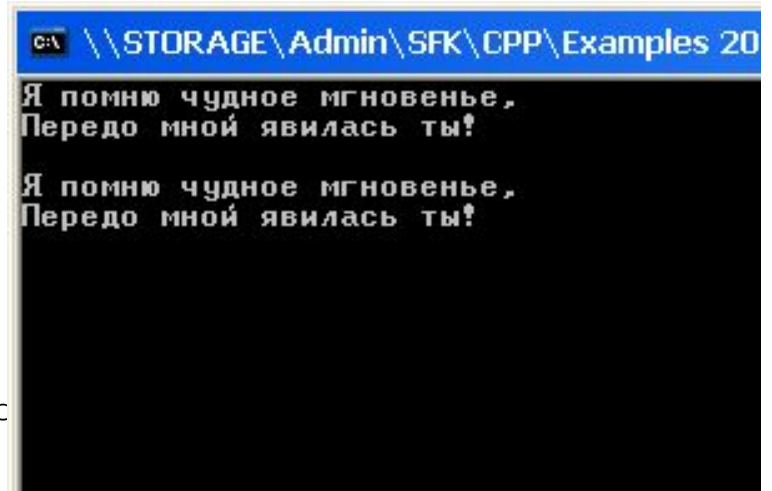
```
for (i=0; i<M; i++)
{ for (j=0; j<N; j++) printf("%6d",a[i][j]);
  printf("\n");
}
```

Динамическое выделение памяти по строку

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX 1000
main()
{
    int n; // Длина текущей строки
    int i=0; // Число строк
    int j; // Для цикла for
    char* t[MAX]; // Массив под указатели на с
    char s[1024]; // Буфер под строку

    do
    { gets(s);
      if ( (n=strlen(s)) && (i<MAX) )
        { t[i]=(char *)malloc(sizeof(char)*(n+1));
          strcpy(t[i++],s);
        }
      else break;
    }
    while (true);

    for (j=0; j<i; j++) puts(t[j]);
}
```



```
C:\ \\ STORAGE\Admin\SFk\CPP\Examples 20
Я помню чудное мгновенье,
Передо мной явилась ты!

Я помню чудное мгновенье,
Передо мной явилась ты!
```