Модуль 8. Строки

Рассматриваются хранение и методы обработки строковой информации, а также стандартные функции для работы со строками

Строки

const char c=’c’; //символ – занимает один байт, его значение не меняется
char a,b;//символьные переменные, занимают по одному байту, значения меняются
const char *s=“Пример строки\n” ;//текстовая константа

Строка - массив символов, заканчивающийся нуль-символом (символ с кодом, равным 0; записывается '\0‘).
“A” – строка (2байта), ‘A’ – символ (1байт)

```
Библиотеки - <string.h>
             <cstring>
             <string>
Ввод-вывод строк - <cstdio>
```

Инициализация строк

```
void main()
{
char s1[10]="string1";
int k=sizeof(s1);
cout<<s1<<"\t"<<k<<endl;
char s2[]="string2";
k=sizeof(s2);
cout<<s2<<"\t"<<k<<endl;
char s3[]={‘s’,’t’,’r’,’i’,’n’,’g’,’3’}
k=sizeof(s3);
cout<<s3<<"\t"<<k<<endl;
char *s4="string4";//указатель на строку, ее нельзя изменить
k=sizeof(s4);
cout<<s4<<"\t"<<k<<endl;
}
```

Результаты:

string1 10 – выделено 10 байтов, в том числе под \0
string2 8 – выделено 8 байтов (7+1байт под \0)
string3 8 – выделено 8 байтов (7+1байт под \0)
string4 4 – размер указателя

Занесение строк в память

char*ss; - описан указатель
ss=”String6”; // неправильно
//память не выделяется , поэтому программа может
// закончиться аварийно!!!

Правильно:
char *sss=new char[10];//выделяем динамическую память
strcpy(sss,”String7”);//копируем строку в память

char str[10] = “Student"; // выделено 10,заполнено 8 байт

char str[] = “Student"; // выделено и заполнено 8 байт

char *str = “Student” // создан указатель на константу

Ввод-вывод строк

int getchar(void) - осуществляет ввод одного символа их входного потока, при этом она возвращает один байт информации (символ) в виде значения типа int. Это сделано для распознавания ситуации, когда при чтении будет достигнут конец файла.

int putchar (int c) – помещает в стандартный выходной поток символ c.

char* gets(char*s) – считывает строку s из стандартного потока до появления символа ‘\n’, сам символ ‘\n’ в строку не заносится.

int puts(const char* s) - записывает строку в стандартный поток, добавляя в конец строки символ ‘\n’, в случае удачного завершения возвращает значение больше или равное 0 и отрицательное значение (EOF=-1) в случае ошибки.

ПРИМЕРЫ ВВОДА-ВЫВОДА СТРОК

1. char s[20];
cin>>s;//ввод строки из стандартного потока
cout<<s;//вывод строки в стандартный поток
Результат работы программы:
При вводе строки “123 456 789”, чтение байтов осуществляется до первого пробела, т. е. в строку s занесется только первое слово строки “123/0”, следовательно, выведется: 123.
2. char s[20];
gets(s);//ввод строки из стандартного потока
puts(s);//вывод строки в стандартный поток
Результат работы программы:
При вводе строки “123 456 789”, чтение байтов осуществляется до символа ‘\n’, т.е. в s занесется строка”123 456 789\n\0”, при выводе строки функция puts возвращает еще один символ ‘\n’, следовательно, будет выведена строка “123 456 789\n\n”.
3. char s[20];
scanf(“%s”,s);//ввод строки из стандартного потока
printf(“%s”,s);//вывод строки в стандартный поток
Результат работы программы:
При вводе строки “123 456 789”, чтение байтов осуществляется до первого пробела,т. е. в строку s занесется только первое слово строки “123/0”, следовательно, выведется: 123. Т. к. s – имя массива, т. е. адрес его первого элемента, операция & в функции scanf не используется.

Основные функции работы со строками

unsigned **strlen**(const char*s);

Вычисляет длину строки s.

int **strcmp**(const char*s1, const char *s2);

Сравнивает строки s1 и s2. Если s1<s2, то результат отрицательный, если s1==s2, то результат равен 0, если s2>s1 – результат положительный.

int **strcnmp**(const char*s1, const char *s2);

Сравнивает первые n символов строк s1 и s2. Если s1<s2, то результат отрицательный, если s1==s2, то результат равен 0, если s2>s1 – результат положительный.

char***strcpy**(char*s1, const char*s2);

Копирует символы строки s1 в строку s2.

char***strncpy**(char*s1, const char*s2, int n);

Копирует n символов строки s1 в строку s2. Конец строки отбрасывается или дополняется пробелами.

char***strcat**(char*s1,const char*s2);

Приписывает строку s2 к строке s1

char***strncat**(char*s1, const char*s2);

Приписывает первые n символов строки s2 к строке s1

Функции преобразования и анализа строк

```
double atof(const char* p);
//преобразует переданную строку в double
int atoi(const char* p);   //преобразует переданную строку в int
long atol(const char* p); //преобразует переданную строку в long
```

Для работы с символами в стандартной библиотеке (заголовочные файлы <ctype.h> и <cctype>) есть следующие функции:

Проверка на принадлежность символа множеству:

isalnum	букв и цифр (A-Z, a-z, 0-9)
isalpha	букв (A-Z, a-z)
iscntrl	управляющих символов (с кодами 0..31 и 127)
isdigit	цифр (0-9)
isgraph	Печатаемых символов, кроме пробела
islower	букв нижнего регистра (a-z)
isprint	Печатаемых символов
ispunct	знаков пунктуации
isspace	символов-разделитетей
isupper	букв верхнего регистра (A-Z)
isxdigit	шестнадцатеричных цифр (A-F, a-f, 0-9) 88

Пример 1. Запрос пароля не более 3 раз

```
#include <stdio.h>
#include <string.h>
int main(){
   char s[5], passw[] = "kuku";
/* Можно -  *passw = "kuku"; */
int i, ok = 0;
   for (i = 0; !ok && i<3; i++){
      printf("\nвведите пароль:\n");
      gets(s);                          if
(strcmp(s,passw)==0 ) ok=1;
   }
   if (ok) printf("\nпароль принят");
   else printf("\nпароль не принят");
}
```

Пример 2. Удалить из строки слов, разделенных пробелами, слова, начинающиеся с цифры

```
#include <stdio.h>
#include <string.h>
void main()
{ char s[250], //исходная строка
w[25], //слово
mas[10][25];//массив слов
puts("\nвведите строку");
gets(s);
int k=0,t=0,i,len,j;
len=strlen(s);
while(t<len)
{
for(j=0,i=t;s[i]!=' ';i++,j++)w[j]=s[i];//формируем слово до пробела
w[j]='/0';//формируем конец строки
strcpy(mas[k],w);//копируем слово в массив
k++;//увеличиваем счетчик слов
t=i+1;//переходим к следующему слову в исходной строке
}
strcpy(s,"");//очищаем исходную строку
for(t=0;t<k;t++)
if(mas[t][0]<'0'&&mas[t][0]>'9')//если первый символ не цифра
{strcat(s,mas[t]);//копируем в строку слово
  strcat(s," ");//копируем в строку пробел
}
puts(s);//выводим результат
}
```

Пример 3. Работа с указателями на строки

```
char src[10], dest[10];
for (int i = 0; i <= strlen(src); i++)
dest[i] = src[i];
```

```
char *src = new char [10];
   char *dest = new char [10], *d = dest;
   cin << src;
   while ( *src != 0) *d++ = *src++;
   *d = 0;
```

Пример 4. Передача строк в качестве параметров функций

Функция поиска заданного символа в строке

```
int find(char *s,char c)
{
for (int I=0;I<strlen(s);I++)
if(s[I]==c) return I;
return –1
}
```

С помощью этой функции подсчитаем количество гласных букв в строке.

```
void main()
{ char s[255];
gets(s)
char gl=”aouiey”;
for(int I=0,k=0;I<strlen(gl);I++)
if(find(s,gl[I])>0)k++;
printf(“%d”,k);
}
```