**Работа № 4.  
Класс «Динамическая строка» и перегрузка операций**

**Теоретические сведения**

**Задание к работе**

**Варианты заданий**

**Цель работы –** изучить методику создания одномерных динамических символьных массивов при помощи конструкторов с захватом динамической памяти и деструкторов для их уничтожения, а так же способа работы со строковыми объектами. Познакомиться с механизмом перегрузки операций.

## Теоретические сведения

Для представления символьной (текстовой) информации  можно использовать символы, символьные переменные и символьные константы.

Символьная константа представляется последовательностью символов, заключенной в кавычки: “Начало строки \n”. В С++ нет отдельного типа для строк. Массив символов - это и есть строка. Количества элементов в таком массиве на один элемент больше, чем изображение строки, т. к. в конец строки добавлен ‘\0’ (нулевой байт).

Присвоить значение массиву символов с помощью обычного оператора присваивания нельзя. Поместить строку в массив можно либо при вводе, либо с помощью инициализации:

char s[] = “ABCDEF”;

Для работы со строками существует специальная библиотека string.h. Примеры функций для работы со строками из библиотеки string.h в таблице 2:

Таблица 2 - Функции работы со строками

|  |  |
| --- | --- |
| Функция | Прототип и краткое описание функции |
| strcmp | int strcmp(const char \*str1, const char \*str2); Сравнивает строки str1 и str2. Если str1< str2, то результат отрицательный, если str1 = str2, то результат равен 0, если str1> str2, то результат положительный. |
| strcpy | char\* strcpy(char\*s1, const char \*s2); Копирует байты из строки s1 в строку s2 |
| strdup | char \*strdup (const char \*str); Выделяет память и перености в нее копию строки str. |
| strlen | unsigned strlen (const char \*str); Вычисляет длину строки str. |
| strncat | char \*strncat(char \*s1, const char \*s2, int kol); Приписывает kol символов строки s1 к строке s2. |
| strncpy | char \*strncpy(char \*s1, const char \*s2, int kol); Копирует kol символов строки s1 в строку s2. |
| strnset | char \*strnset(char \*str, int c, int kol); Заменяет первые kol символов строки s1  символом с. |

Строки, при передаче в функцию, в качестве фактических параметров могут быть определены либо как одномерные массивы типа char[], либо как указатели типа char\*.  В отличие от обычных массивов в этом случае нет необходимости явно указывать длину строки.

Функции преобразования строки S в число:

целое:  int atoi(S); длинное целое:  long atol(S); действительное:  double atof(S); при ошибке возвращает значение 0.

Функции преобразования числа V в строку S:

целое:  itoa(int V,char S,int kod); длинное целое:  ltoa(long V,char S,int kod); 2<=kod<=36, для отрицательных чисел kod=10.

Перегрузка операций

Для перегрузки операции для класса в С++ используется следующий синтаксис:

<Тип>  operator <операция>(<входные параметры>){   <операторы>;}

где < Тип > - тип, возвращаемый функцией;

operator - ключевое слово;

< операция > - перегружаемая операция.

В языке С++ имеются следующие ограничения на перегрузку операций:

* С++ не различает префиксную и постфиксную формы ++ и - -;
* переопределяемая операция должна присутствовать в языке (например, нельзя определить операцию с символом #);
* нельзя переопределить операторы, заданные следующими символами . \* :: ?  ;
* переопределённые операции сохраняют свой изначальный приоритет.

Наличие в классе конструктора String:: String(String&) и операторов присваивания позволяет защитить объекты класса от побитового копирования.

Пример**:** Ввести с клавиатуры строку символов. Признак окончания ввода строки - нажатие клавиши "Ввод". Программа должна определить длину введенной строки L и если L<10 – возвратить строку, которая не содержит  заглавных латинских букв.

#include <iostream.h>  
#define SIZE 255 //длина строки по умолчанию  
#include <string.h>  
#include <stdlib.h>  
#include <stdio.h>  
#include <istream.h>

class X{  
char \*str;   
char \*str\_return;  
public:  
X(); //конструктор по-умолчанию  
X(char\*); //конструктор, которому можно передавать параметр  
~X(); //деструктор  
char\* Run(); //метод, выполняющий поставленную задачу.   
void Set(char\*);  
friend void print(X&); //функция-друг печати  
friend ostream& operator<<(ostream&,X&); //перегрузка оператора вывода  
friend istream& operator>>(istream&,X&); //перегрузка оператора ввода  
friend char\* Run(X&); //функция-друг, выполняющий поставленную задачу.

};

X::X(){  
str=new char[SIZE];   
str[0]='\0';          
str\_return=new char[SIZE];  
str\_return[0]='\0';  
};  
X::X(char \*s){  
str=new char[SIZE];  
strcpy(str,s);  
str\_return=new char[SIZE];  
str\_return[0]='\0';  
};  
X::~X(){  
delete[] str;  
cout<<"...destructor has been called"<<endl;  
};

void X::Set(char\* s){  
for (unsigned int i=0;i<strlen(s);i++)   
str[i]=s[i];  
str[i]='\0';  
};

char\* X::Run(){ /\*метод, решающий конкретную задачу, в данном случае - выделение из строки подстроки, не содержащей заглавных латинских букв, если длина исходной строки меньше 10\*/  
int j=0;   
if (strlen(str)<10) {  
for (unsigned int i=0;i<strlen(str);i++)  
if ( ((int)str[i]<65) || ((int)str[i]>90) ) {  
str\_return[j]=str[i]; j++;  
};  
str\_return[j]='\0';  
}  
else strcpy(str\_return,str);  
  
return str\_return;    
};

char\* Run(X &obj){return obj.Run();};

void print(X &obj){cout<<obj.str<<" "<<obj.str\_return<<endl;};

ostream& operator<<(ostream &stream,X &ob) {  
stream << ob.str ;   
return stream;  
};

istream &operator>>(istream &stream,X &ob){  
stream >> ob.str;  
return stream;  
};  
void main (void){  
char s[265];  
  
cout<<"Type anything and press \"Enter\":"<<endl;  
cin.getline(s,256); //считываем полностью всю строку  
X str(s); //доступ к методам класса непосредственно через переменную,  
//начальное значение устанавливаем через конструктор  
cout<<"You have type:"<<endl;  
print(str);  
cout<<"Output string:"<<endl;  
cout<<Run(str)<<endl;   
cout<<"Type anything and press \"Enter\":"<<endl;  
cin.getline(s,256);  
X \*pstr; //доступ к методам класса через указатель  
pstr=new X();  
pstr->Set(s);  
cout<<"You have type:"<<endl;  
print(\*pstr);  
cout<<"Output string:"<<endl;  
cout<<Run(\*pstr)<<endl;  
delete pstr;

};

## Задание к работе

Общая постановка. Пользовательский класс String должен содержать необходимые элементы-данные, которые создаются в динамической области памяти.

* Конструктор для создания строк: String (…);
* Деструктор: ~String();
* Метод ввода исходной строки: Set();
* Метод печати: void print(…);

Код методов  – вне пространства определения класса. Программа иллюстрирует прямой и косвенный способы обращения к методам.

Ввести с клавиатуры строку  символов S1. Признак окончания ввода строки - нажатие клавиши "Ввод". Программа должна содержать перегруженную операцию **«=»**, использование которой скопирует S1 в S2 .

Исходную и преобразованную строки вывести в файл.

## Варианты заданий

1. Длина L нечетная, то удаляется символ, стоящий посередине строки;