

```

#include <iostream.h>
#include <string.h>

extern void exit(int);

class string {
    struct srep {
        char* s;
        int n;
    };
    srep *p;

public:
    string(char *);
    string();
    string(string &);
    string& operator=(char *);
    string& operator=(string &);
    ~string();
    char& operator[(int i)];

    friend ostream& operator<< (ostream&, string&);
    friend istream& operator>> (istream&, string&);

    friend int operator==(string &x, char *s)
        { return strcmp(x.p->s, s) == 0; }

    friend int operator==(string &x, string &y)
        { return strcmp(x.p->s, y.p->s) == 0; }

    friend int operator!=(string &x, char *s)
        { return strcmp(x.p->s, s) != 0; }

    friend int operator!=(string &x, string &y)
        { return strcmp (x.p->s, y.p->s) != 0; }
};

string::string()
{
    p = new srep;
    p->s = 0;
    p->n = 1;
}

```

```

string::string(char* s)
{
    p = new srep;
    p->s = new char[ strlen(s) +1];
    strcpy(p->s, s);
    p->n = 1;
}

```

```

string::string(string& x)
{
    x.p->n++;
    p = x.p;
}

```

```

string::~~string()
{
    if (--p->n == 0){
        delete p->s;
        delete p;
    }
}

```

```

string& string::operator=(char* s)
{
    if (p->n > 1) {
        p->n--;
        p = new srep;
    }
    else if (p->n == 1)
        delete p->s;

    p->s = new char[ strlen(s)+1 ];
    strcpy(p->s, s);
    p->n = 1;
    return *this;
}

```

```

string& string::operator=(string& x)
{
    x.p->n++;
    if (--p->n == 0) {
        delete p->s;
        delete p;
    }
    p = x.p;
    return *this;
}

ostream& operator<<(ostream& s, string& x)
{
    return s << x.p->s << " [" << x.p->n << "]\n";
}

istream& operator>>(istream& s, string& x)
{
    char buf[256];
    s>>buf;
    x = buf;
    cout << "echo: " << x << "\n";
    return s;
}

void error(char* p)
{
    cout << p << "\n";
    exit(1);
}

char& string::operator[](int i)
{
    if (i<0 || strlen(p->s)<i)
        error("index out of range");
    return p->s[i];
}

```

```

main()
{
    string x[100];
    int n;

    cout << "here we go\n";

    for (n = 0; cin>>x[n]; n++) {
        string y;
        if (n==100) error("too many strings");
        cout << (y = x[n]);
        if (y=="done") break;
    }

    cout << "here we go back again\n";
    for (int i=n-1; 0<=i; i--) cout << x[i];
}

```

```

.....
.....
.....

```