#define DIM(a) sizeof(a)/sizeof(\*a)

//交换

#define SWAP(p,q) \

do { \

\*p^=\*q; \

\*q^=\*p; \

\*p^=\*q; \

} while(0)

//一行代码求字符串长度(函数版本)

int str\_len(const char\* s)

{

//assert函数　assert的作用是先计算表达式 expression ，

//如果其值为假（即为0）打印一条出错信息，然后通过调用 abort 来终止程序运行。

//return s, (\*s ? str\_len(s + 1) + 1 : 0);

return assert(s), (\*s ? str\_len(s + 1) + 1 : 0);//用assert对NULL报错

}

//逆序

char\* strrev(char \* str)

{

int i,len;

for(i=0,len=strlen(str);i<len/2;i++)

{

SWAP(&str[i],&str[len-i-1]);

}

return str;

}

//字符串拷贝

char \*Strcpy(char \*dest,const char \*source)

{

if(NULL==dest||NULL== source) return NULL;

else

{

char \*dest\_save=dest;

//一行代码完成字符串拷贝

while((\*dest++=\*source++)!='\0');

return dest\_save;

}

}

//字符串衔接

char \*Strcat(char \*dest,const char \*source)

{

if(dest)

{

char \*p=dest;

while(\*p!='\0')

++p;

Strcpy(p,source);

return dest;

}else return NULL;

}

//字符比较

int Strcmp(const char \*s1,const char \*s2)

{

while (\*s1!='\0'&&\*s2!='\0')

if(\*s1++!=\*s2++) break;

return (\*--s1 - \*--s2);

}

//搜索字符串特定字符第一次出现的位置！！！！！

int Strchr(const char \*sourcre,int n,char ch)

{

if(NULL==sourcre||n<0) return -1;

else

{

int i;

for(i=0;i<n&&\*(sourcre+i)!=ch;++i);

if(i==n) return -1;

return (i+1);

}

}

//内存拷贝覆盖

void \*Memcpy(void \*memTo,const void \*memFrom,size\_t size)//size\_t=long unsigned int(32bit)

{

char \*tmpTo=(char\*)memTo;

const char\*tmpFrom = (const char\*)memFrom;

if(NULL==memTo||NULL==memFrom||size<=0) return memTo;

while (size--) \*tmpTo++=\*tmpFrom++;

return memTo;

}

//字符循环右移

char\* right\_shift\_r(const char\* src,unsigned int n)

{

char\* result;

const unsigned int LEN = strlen(src);//没有记录\0

int i = 0;

for(i=0; i < LEN; i++)

{

result[(n + i) % LEN] = src[i];//O(len)

}

//我们处理的是字符数组而不是字符串，且\0永远在末尾！！！！

result[LEN] = '\0';//添加边界！！！！！！！

return result;

}

int str2int(const char \*str)

{

int res,i;

res=i=0;

int bPositive=1;//正负号

if('-'==str[0])//判定符号

{

bPositive=0;

++i;

}

while (str[i]!='\0')

{

//res=res\*10;一样的

res=(res<<1)+(res<<3);//左移理解为乘法2^n 这里res\*10

res+=str[i++]-'0';//减去'0'

}

if(!bPositive)

res=-res;

return res;

}

void int2str(int n,char \*str)

{

int j;

if(NULL==str)

printf("ERROR!\n");

//加负号

j=0;

if(n<0)

{

n=-n;

str[j++]='-';

}

while (n)

{

str[j++]=n%10+'0';//取余 从最低位取值

n/=10;

}

str[j]='\0';

if('-'==str[0])//颠倒数字字符串

{

strrev(str+1); //别把负号搅进去了

}else

{

strrev(str);

}