

String Functions

General Functions	
type	Data type in string:char-0,float-1,integer-2
isnum	Char is:alfa(0),digit(1)
isupp	Char is:uppercase alphabet(1)
islow	Char is:lowercase alphabet(1)
isalf	Char is:alphabet(1)
ltou	Convert lowercase alphabet to uppercase alphabet
utol	Convert uppercase alphabet to lowercase alphabet
len	length of char string
Cjust	Center justify source string to destination string(new string of specified/source length or specified string)
Ljust	Left justify source string to destination string(new string of specified/source length or specified string)
Rjust	Right justify source string to destination string(new string of specified/source length or specified string)
ltrim	Remove leading blanks from string.
rtrim	Remove trailing blanks from string.
atrim	Remove leading/trailing blanks from string.
upper	Convert String to uppercase
join	Join string str2 or char to end of str1.
subs	Extract substring from string.
asc2uni	Convert ASCII(8 bit) string to new or specified destination UNICODE(16 bit) string
scopy	Copy string to new or specified destination string
scmp	Compare strings
at	Position of first occurrence of str2 or char in str1
rat	Position of last occurrence of str2 or char in str1
empty	Determine if string is empty
repl	Fill 1st 'n' positions of new or specified string w/ char 'c'
spac	Fill 1st 'n' positions of new or specified string w/ spaces
pad	Prefix i/p string with spaces
GetStr	Extract next 'chr' delimited substring from string
PutStr	Convert 2D string to a 1D string of 'chr' delimited substrings
array	Position of framing character(chr) in string(str1)
FrmStr	Extract left(lchr) & right(rchr) framing characters from string
Del2d	delete 2d string

Math Functions	
val	Convert string to double precision float number. if i/p is !numeric, returns 0
str	Convert double precision float number to string
itos	Convert unsigned integer to string
dnegh	signed integer number to hex string
dnegb	signed integer number to binary string
itoh	unsigned integer number to hex string
dtob	unsigned integer number to binary string
bnegd	binary string(16 char max) to signed integer number
btod	binary string(16 char max) to unsigned integer number
htoi	hex string(4 char max) to unsigned integer number
rtoh	32 bit float number to 4 char hex string in real4 format
htor	4 char hex string in real4 format to 32 bit float number
htob	hex string to binary string
btoh	binary string to hex string

Common Function Inputs

- p* 0-*str1*[] not dynamic or dynamic but not deleted. *str2*[] not dynamic or dynamic but not deleted.
1-*str1*[] is dynamic and deleted in function. *str2*[] not dynamic or dynamic but not deleted.
2- *str1*[] not dynamic or dynamic but not deleted. *str2*[] is dynamic and deleted in function.
3- *str1*[] is dynamic and deleted in function. *str2*[] is dynamic and deleted in function.
- str1* Pointer to a NULL terminated ASCII(8bit) source string. Is 'dynamic' if created with the 'new' operator. If 'dynamic' it must be explicitly deleted in any function with access to its address
- str2* Pointer to a NULL terminated ASCII(8bit) source string. Is 'dynamic' if created with the 'new' operator. If 'dynamic' it must be explicitly deleted in any function with access to its address
- c* ASCII character eg 'a', 'b', '1', '.', '/', '\\'(for '\\').
- dtr* Pointer to a NULL terminated ASCII(8bit) string, specified to store the result. Can't be a constant string. If it is 'dynamic', the calling program must ensure deletion.
- cse*n Comparison is case sensitive(1) or not(0)
- num* Signed integer number to be converted
- inum* Unsigned integer number to be converted
- snum* Binary number to be converted, represented as a string of binary digits
- hnum* Hex number to be converted, represented as a string of case nsensitive hex digits
- inv* Bits of the output should be inverted(1) or not(0)
- wd* Width of the output if it were a string of binary digits
If *wd*>0 & *wd*>natural length the leading spaces are filled w/ '0'
If *wd*>0 & *wd*<natural length, LSBs digits are truncated
If *wd*=0 natural length is returned

*Note:-

A numeric string argument may have leading and/or trailing spaces. It may be prefixed with a single sign character. After the optional sign character it can't contain embedded non digits. All other strings are character strings.

String	type	String	type
"abc"	char	"+0.123"	float numeric
"12 3"	char	" +1.234 "	float numeric
"a123"	char	"-123"	integer numeric
"+ 123"	char	"123"	integer numeric
".123"	float numeric	" 1234 "	integer numeric

General Functions

type

DWORD type(char *str1)

Description

Returns the type of the specified character expression.

Returns

Argument type	return
Character(non-numeric) string	0
Floating point numeric string	1
Fixed point(integer) numeric string	2

isnum

DWORD isnum(char c)

Description

Returns 1 if the argument is a digit between zero and nine otherwise returns 0

isupp

DWORD isupp(char c)

Description

Returns 1 if the argument is an uppercase alphabet otherwise returns 0

islow

DWORD islow(char c)

Description

Returns 1 if the argument is an lowercase alphabet otherwise returns 0

isalf

DWORD isalf(char c)

Description

Returns 1 if the argument is an uppercase or lowercase alphabet otherwise returns 0.

ltou

char ltou (char c)

Description

Convert alphabet to uppercase. If argument is not an alphabet, it is returned unchanged.

utol

char utol (char c)

Description

Convert alphabet to lowercase. If argument is not an alphabet, it is returned unchanged.

len

DWORD len(char *str1,DWORD p)

Description

Returns the length of a character string. Each byte counts as one and counting stops at the NULL termination and the NULL character is not included in the count returned.

Examples

```
len("123"); //Returns:3  
len(""); //Returns:0  
len(0); //Returns:0
```

Cjust

#	Function Prototype	Description
1	<code>char * Cjust(char *str1,DWORD ln,DWORD p)</code>	Center justify source string to new result string of specified length
2	<code>char * Cjust(char *str1,DWORD p)</code>	Center justify source string to new result string
3	<code>char * Cjust(char *dtr,char *str1,DWORD p)</code>	Center justify source string to specified result string

1>The justified result is placed in a **new result string** of length $\langle ln \rangle$. If $\langle ln \rangle$ is less than the length of the source string, the RHS and LHS of the result are truncated equally.

2>The justified result is placed in a **new result string** of length equal to the length of the source.

3>The justified result is placed in the **specified result string**. if the length of this string is less than the length of the source, the RHS and LHS of the result are truncated equally.

Returns

In function 3>, the address of the **specified result string**. In functions 1> and 2>, the Address of dynamic **new result string**, used by the calling function to delete it.

Ljust

#	Function Prototype	Description
1	<code>char * Ljust(char *str1,DWORD ln,DWORD p)</code>	Left justify source string to new result string of specified length
2	<code>char * Ljust(char *str1,DWORD p)</code>	Left justify source string to new result string
3	<code>char * Ljust(char *dtr,char *str1,DWORD p)</code>	Left justify source string to specified result string

1>The justified result is placed in a **new result string** of length $\langle ln \rangle$. If $\langle ln \rangle$ is less than the length of the source string, the RHS of the justified result is truncated. Otherwise, it is padded with spaces.

2>The justified result is placed in a **new result string** of length equal to the length of the source.

3>The justified result is placed in the **specified result string**. if the length of this string is less than the length of the source, the RHS of the result is truncated. Otherwise, it is padded with spaces.

Returns

In function 3>, the address of the **specified result string**. In functions 1> and 2>, the Address of dynamic **new result string**, used by the calling function to delete it.

Rjust

#	Function Prototype	Description
1	<code>char * Rjust(char *str1,DWORD ln,DWORD p)</code>	Right justify source string to new result string of specified length
2	<code>char * Rjust(char *str1,DWORD p)</code>	Right justify source string to new result string
3	<code>char * Rjust(char *dtr,char *str1,DWORD p)</code>	Right justify source string to specified result string

1>The justified result is placed in a **new result string** of length $\langle ln \rangle$. If $\langle ln \rangle$ is less than the length of the source string, the LHS of the justified result is truncated. Otherwise, it is padded with spaces.

2>The justified result is placed in a **new result string** of length equal to the length of the source.

3>The justified result is placed in the **specified result string**. if the length of this string is less than the length of the source, the RHS of the result is truncated. Otherwise, it is padded with spaces.

Returns

In function 3>, the address of the **specified result string**. In functions 1> and 2>, the Address of dynamic **new result string**, used by the calling function to delete it.

ltrim

char * ltrim(char *str1,DWORD p)

Description

Remove leading blanks from source string to a **new result string**. If the source string is a null string("") or all spaces, the result is a null("") string.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

rtrim

char * rtrim(char *str1,DWORD p)

Description

Remove trailing blanks from source string to a **new result string**. If the source string is a null string("") or all spaces, the result is a null("") string.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

atrim

char * atrim(char *str1,DWORD p)

Description

Remove leading & trailing blanks from source string to a **new result string**. If the source string is a null string("") or all spaces, the result is a null("") string.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

upper

char * upper(char *str1,DWORD p)

Description

Convert source string to uppercase in a **new result string** with length of the source string.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

join

#	Function Prototype	Description
1	char * join(char* str1,char* str2,DWORD p)	Join string str2 to end of str1
2	char * join(char *str1,char chr,DWORD p)	Join 'char' to end of str1

Used to concatenate strings to a **new result string** with length equal to the combined length of the input strings.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

subs

char * subs(char *str1,int w,int d,DWORD p)

Description

Returns **d** characters of string from & including the **wth** character(character number starts from one for the left most character) in a **new result string**. Depending on the values of **d** and **w** leading and trailing spaces are inserted into the returned string:-

d<0 or (**d=0 & w>=0**)

A NULL new string is returned

d>0 & w>=0

d elements of **str1** starting from the **wth** (**w=0** is taken as 1) are padded with trailing spaces to the extent that **d** exceeds the remaining length of **str1** and returned in a new string.

d>=0 & w<0

d elements of **str1** starting from the 1st are padded with trailing spaces to the extent that **d** exceeds the length of **str1**, padded with 0-**w** leading spaces and returned in a new string.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

Examples

```
char *a=subs("abcdef",9,3); //Returns: "..."  
char *a=subs("abcdef",6,3); //Returns: "f.."  
char *a=subs("abcdef",-1,3); //Returns: ".abc"  
char *a=subs("abcdef",-1,6); //Returns: ".abcdef.."  
char *a=subs("abcdef",-1,0); //Returns: ". "  
char *a=subs("abcdef",2,0); //Returns: ""  
char *a=subs("abcdef",8,0); //Returns: ""  
char *a=subs("abcdef",0,3); //Returns: "abc"  
char *a=subs("abcdef",1,3); //Returns: "abc"
```

Note: The '.' character shown above represents a space character

asc2uni

#	Function Prototype	Description
1	WORD *asc2uni(char *str1)	Convert ASCII(8 bit) string to new result UNICODE(16 bit) string
2	WORD *asc2uni(WORD *dtr,char *str1,DWORD p)	Convert ASCII(8 bit) string to specified result UNICODE(16 bit) string

Parameters

dtr Pointer to a NULL terminated UNICODE(16bit) string, specified to store the result. Can't be a constant string. If it is 'dynamic', the calling program must ensure deletion.

Returns

In function 2>, the address of the **specified result string**. In functions 1>, the Address of dynamic **new result string**, used by the calling function to delete it.

scopy

#	Function Prototype	Description
1	char *scopy(char *dtr,char *str1,DWORD p)	Copy source string to specified result string
2	char *scopy(char *str1,DWORD p)	Copy source string to new result string

Returns

In function 2>, the address of the **specified result string**. In functions 1>, the Address of dynamic **new result string**, used by the calling function to delete it.

scmp

#	Function Prototype	Description
1	DWORD scmp(char *str1,char *str2,DWORD p)	Compare strings
2	DWORD scmp(char *str1,char *str2,DWORD exact,DWORD csen, DWORD p)	Compare strings

Compare the input strings depending on the conditions specified by the *csen* and *exact* inputs.

Parameters

<i>exact</i>	Condition
0	Exclude trailing spaces and ignore extra length of <i>str1</i>
1	Exclude trailing spaces and don't ignore extra length of <i>str1</i>
2	Include trailing spaces and don't ignore extra length of <i>str1</i>
3	Include trailing spaces and ignore extra length of <i>str1</i>

Returns

1-Result of comparison is TRUE
0-Result of comparison is FALSE

at

#	Function Prototype	Description
1	DWORD at(char chr,char *str1,DWORD csen, DWORD p)	Determine the position of the first occurrence of character <i><chr></i> in string <i><str1></i>
2	DWORD at(char *str2,char *str1,DWORD csen, DWORD p)	Determines the position of the first occurrence of substring <i><str2></i> in string <i><str1></i>

Returns

The position of the first instance of the LHS argument:character *<chr>*(func 1>) or *<str2>*(func 2>) within the RHS argument:string *<str1>* as a integer numeric value. If not found or the RHS argument is zero(NULL) at() returns zero.

rat

#	Function Prototype	Description
1	DWORD rat(char chr,char *str1,DWORD csen, DWORD p)	Determine the position of the last occurrence of character <i><chr></i> in string <i><str1></i>
2	DWORD rat(char *str2,char *str1,DWORD csen, DWORD p)	Determines the position of the last occurrence of substring <i><str2></i> in string <i><str1></i>

Returns

The position of the last instance of the LHS argument:character *<chr>*(func 1>) or *<str2>*(func 2>) within the RHS argument:string *<str1>* as a integer numeric value. If not found or the RHS argument is zero(NULL) at() returns zero.

empty

DWORD empty(char *str1,DWORD p)

Description

Determines if the input string contains non-space characters

Returns

1-If the input string contains only space characters or is a NULL string("").
0-If the input string contains non-space characters.

repl

#	Function Prototype	Description
1	<code>char *repl(char c,DWORD n)</code>	Create new result string of length(n) filled w/ char 'c'
2	<code>char *repl(char *dtr,char c,DWORD n)</code>	Fill 1st 'n' places of specified result string w/ char 'c'. If the specified result string is longer than <n>, it is truncated.

Returns

In function 2>, the address of the **specified result string**. In functions 1>, the Address of dynamic **new result string**, used by the calling function to delete it.

spac

#	Function Prototype	Description
1	<code>char * spac(DWORD n)</code>	Create new result string of length(n) filled w/ spaces
2	<code>char * spac(char *dtr,DWORD n)</code>	Fill 1st 'n' positions of specified result string w/ spaces. If the specified result string is longer than <n>, it is truncated.

Returns

In function 2>, the address of the **specified result string**. In functions 1>, the Address of dynamic **new result string**, used by the calling function to delete it.

pad

`char *pad(DWORD w,char *str1,DWORD p)`

Description

Prefix i/p string with <w> spaces and return in a **new result string**.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

GetStr

`char * GetStr(char *str1,DWORD beg,DWORD trm,char c)`

Parameters

beg Number-1 of the <chr> delimited substring to be extracted from string <str1>.

trm 1-Remove leading and trailing spaces from the **new result string** returned.

0-Don't remove leading and trailing spaces from the **new result string** returned.

Description

Extract <chr> delimited substring number <beg>+1 from string <str1> & return in **new result string**.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

Example

```
char *str1="adc, def, ghi";
```

```
char *txt[5];
```

```
DWORD i;
```

```
for(i=0; *(txt[i]=GetStr(str1,i,1,','))!=0;i++);
```

```
//Returns:txt[0]="adc", txt[1]=" def", txt[2]=" ghi", txt[3]=""
```

PutStr

char * PutStr(char ** txt,char c)

Parameters

txt Pointer to 2D string(array of 1D strings), the last of which is a NULL string("").

Description

Convert a 2D string <*txt*> to a series of character <*c*> delimited substrings in a 1D **new result string**

Example

```
char *str1[]={ "abc","def",""};
char *txn=PutStr(str1) //Returns: "abc,def"
```

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

array

DWORD array(char *str1,char c)

Description

Returns the position of 1st occurrence of character<*c*>, in string<*str1*>.

Used to detect the start/end(framing characters) of a series of delimited substrings grouped as an array within strings.

Note

The character<*c*> must be preceded or succeeded with either spaces or nothing to be detected.

eg In " [abc" or "abc" " or "[abc" or "abc]". '[' and ']' are detected.

In "a[bc" or "ab]c". '[' and ']' are not detected.

Returns

Position of 1st occurrence of character<*c*>, in <*str1*> as a integer number. If not found, returns zero.

Example

```
DWORD x=array("[abc,def]",'['); //Returns: 1
```

FrmStr

char * FrmStr(char *str1,char lchr,char rchr,DWORD p)

Parameters

lchr Start framing character, of a series of delimited substrings grouped as an array within <*str1*>

rchr End framing character, of a series of delimited substrings grouped as an array within <*str1*>

Description

Extract <*lchr*> and <*rchr*>, framing characters, of a series of delimited substrings grouped as an array within string <*str1*>. Return the string with the framing characters extracted in a **new result string**.

Returns

Address of dynamic **new result string**, used by the calling function to delete it.

Example

```
char *x=FrmStr(" [ abc,def ] "); //Returns: " abc,def "
```

Del2d

void Del2d(char **txt,DWORD flds)

Parameters

flds Number of 1D strings in the 2D string input <*txt*>

Description

delete dynamic 2D string.

Math Functions

val

double val(char *str1,DWORD p)

Description

Convert double precision float number represented as decimal digits in input string to double precision float number. Returns 0 if i/p is not valid numeric string.

itos

char * itos(int n,DWORD w)

Description

Called by str() to convert unsigned integer numbers to decimal digits in a character string.

Parameters

n Unsigned integer number to be converted to a character string.

w Length of the character string to return including sign.

Returns

Address of dynamic **new result string** of length $\langle w \rangle$, used by the calling function to delete it.

Notes

- If *w* is greater than required, the integer is right justified(spaces on the LHS)
- If *w* is less than required, the MSB digits of the integer are truncated.
- If $w=0$ a NULL string is returned
- If $w=1$ and $n<0$, "-" is returned

Examples

```
char *a=itos(-123,4);           //Returns: "-123"  
char *a=itos(-123,3);           //Returns: "-23"  
char *a=itos(-123,1);           //Returns: "-"  
char *a=itos(-123,0);           //Returns: ""
```

str

#	Function Prototype	Description
1	char * str(double n,DWORD w,DWORD d)	Convert float to string of length <i>w</i>
2	char * str(double n,DWORD w)	Convert float to string of length <i>w</i> , $d=0$
3	char * str(double n)	Convert float to string of length=# of integer digits, $d=0$ (length is incremented by one for '-' sign if required)

Convert double precision float numbers to decimal digits in a character string. Strings are formatted using the $\langle w \rangle$ & $\langle d \rangle$ inputs. The inverse of str() is val() which converts character strings to numbers

Parameters

n double precision floating point number(8 byte) to be converted to a character string.

d Number of places after the decimal in the result.

w Length of the character string to return including decimal digits, decimal point and sign.

Returns

Address of dynamic **new result string** of length $\langle w \rangle$, used by the calling function to delete it.

Notes

- If $d>0$, number of places before the decimal in the result is: $i=w-d-1$
- If $d=0$, number of places before the decimal in the result is: $i=w$
- If *d* is less than the number of decimal digits, the LSB digits of the decimal are truncated
- If *d* is zero the return string is truncated to an integer
- If $i<0$ and $n \geq 0$, the decimal can't be placed so spaces are returned
- If $i<1$ and $n < 0$, the '-' sign can't be placed so spaces are returned
- If *i* is greater than required, the integer part is right justified(spaces on the LHS)
- If *i* is less than required, the MSB digits of the integer are truncated.

Examples

Func Call(-ive)	Return	Func Call(+ive)	Return
str(-123.456,8,3)	"-123.456"	str(123.456,8,3)	" 123.456"
str(-123.456,7,3)	"-23.456"	str(123.456,5,3)	"3.456"
str(-123.456,5,3)	"-.456"	str(123.456,4,3)	".456"
str(-123.456,4,3)	" "	str(123.456,3,2)	".45"
str(-123.456,2,0)	"-3"	str(123.456,3,3)	" "
str(-123.456,1,0)	"-"		

dnegh

char *dnegh(int num,DWORD wd)

Description

signed integer number to sign extended hex string

Returns

Address of dynamic **new result string** of length $\langle wd \rangle / 4$ (rounded to the next whole number), used by the calling function to delete it.

Examples

```
char *a=dnegh(-103,8);           //Returns: "99"
```

dnegb

char *dnegb(int num,DWORD wd)

Description

signed integer number to sign extended binary string.

Returns

Address of dynamic **new result string** of length $\langle wd \rangle$, used by the calling function to delete it.

Examples

```
char *b=dnegb(-103,8);           //Returns: "10011001"
```

itoh

char *itoh(DWORD inum,DWORD wd,DWORD inv)

Description

unsigned integer number to hex string.

Returns

Address of dynamic **new result string** of length $\langle wd \rangle / 4$ (rounded to the next whole number), used by the calling function to delete it.

Examples

```
char *c=itoh(103,8,0);           //Returns: "67"  
char *d=itoh(103,8,1);           //Returns: "98"
```

dtob

char *dtob(DWORD inum,DWORD wd)

Description

unsigned integer number to binary string.

Returns

Address of dynamic **new result string** of length $\langle wd \rangle$, used by the calling function to delete it.

Examples

```
char *e=dtob(103,8);           //Returns: "01100111"
```

bnegd

int bnegd(char *snum,DWORD dwd,DWORD p)

Description

binary string(16 char max) to signed integer number.

Parameters

dwd Character position, from RHS, in <*snum*>, of the 1st sign bit, shown below in green.

Returns

Input string converted to signed integer number.

Examples

```
int f =bnegd("010110",6,0);           //Returns: "22.0"  
int g=bnegd("101010",6,0);           //Returns: "-22.0"
```

btod

DWORD btod(char *snum,DWORD inv,DWORD p)

Description

binary string(16 char max) to unsigned integer number.

Returns

Input string converted to unsigned integer number.

Examples

```
DWORD h=btod("010110",0,0);           //Returns: "22.0"
```

htoi

DWORD htoi(char *hnum,DWORD inv,DWORD p)

Description

hex string(4 char max) to unsigned integer number.

Returns

Input string converted to unsigned integer number.

Examples

```
DWORD i=htoi("7F",0,0);               //Returns: "127"
```

rtoh

char *rtoh(char *dnum,DWORD inv,DWORD p)

Description

32 bit float number represented as decimal digits in the input string to 4 char hex string in real4 format. The input string must have a decimal point and at least one decimal digit for correct operation

Parameters

dnum 32 bit float number represented as decimal digits in a character string

Returns

Address of dynamic **new result string** of length <*w*>,used by the calling function to delete it.

Examples

```
char *j=rtoh("103.0",0,0);             //Returns: "43670000"  
char *k=rtoh("-103.0",0,0);           //Returns: "C3670000"
```

htor

double htor(char *rnum,DWORD p)

Description

32 bit float number, in real4 format, represented as hex digits(4) in the input string to float number.

Parameters

rnum 32 bit float number, in real4 format, represented as hex digits(4) in a character string.

Returns

Input string converted to a sign omitted float number.

Examples

```
double l=htor("43670000",0);           //Returns: "103.0"  
double m=htor("C3670000",0);           //Returns: "103.0". This is "-103.0" with '-' sign omitted
```

htob

char* htob(char *hnum,DWORD p)

Description

hex string to binary string.

Returns

Address of dynamic **new result string** of length=len(*hnum*)*4,used by the calling function to delete it.

Examples

```
char *n=htob("7f",0);                 //Returns: "01111111"  
char *o=htob("07f",0);                //Returns: "00000111111111"
```

btoh

char* btoh(char *snum,DWORD inv,DWORD p)

Description

binary string to hex string.

Returns

Address of dynamic **new result string** of length <w>,used by the calling function to delete it.

Examples

```
char *p=btoh("01111111",0,0);         //Returns: "7f"  
char *q=btoh("0011111111",0,0);      //Returns: "07f"
```