

Important functions in the stdlib.h library of C

🕒 Oct. 18, 2017 🌐 C (/blog/tag/c/?tag=c) C++ (/blog/tag/cpp/?tag=cpp) EXAMPLE (/blog/tag/example/?tag=example) FUNCTION (/blog/tag/function/?tag=function) 🔍 10299

✍️

Become an Author

(/blog/submit-article/)

🔍 ×

Ad - spinny.com

More ▼

This post lists and explains some of the important functions of the stdlib.h library of C.

| Function | Description |
|----------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| atof | converts string to float |
| atoi | converts string to integer |
| atol | converts string to long int |
| atoll | converts string to long long int |
| strtod | converts string to double |
| strtof | converts string to float |
| strtol | converts string to long int |
| strtold | converts string to long double |
| strtoll | converts string to long long int |
| strtoul | converts string to unsigned long int |
| strtoull | converts string to unsigned long long int |
| rand | returns a pseudo-random integer |
| srand | seeds the rand function |
| bsearch | performs binary search on an array |
| qsort | performs quicksort on an array |
| abs | returns absolute value |
| div | performs division |
| labs | returns absolute value of long int |
| ldiv | performs division |
| llabs | returns absoulte value of long long int |
| lldiv | performs division |
| abort | terminates the program |
| atexit | calls a function at the exit of the program |
| at_quick_exit | executes when quick_exit is called |
| exit | terminates a program |
| getenv | returns the enviornment of the operating system |
| quick_exit | terminates the program |
| system | used to run another program |
| _EXIT | terminates a program |
| calloc (https://www.codesdope.com/c-dynamic-memory/) | allocates (https://www.codesdope.com/c-dynamic-memory/)memroy (https://www.codesdope.com/c-dynamic-memory/) at the runtime (https://www.codesdope.com/c-dynamic-memory/) |
| free (https://www.codesdope.com/c-dynamic-memory/) | deallocates the memory (https://www.codesdope.com/c-dynamic-memory/) |
| malloc (https://www.codesdope.com/c-dynamic-memory/) | allocates memory at the runtime (https://www.codesdope.com/c-dynamic-memory/) |
| realloc (https://www.codesdope.com/c-dynamic-memory/) | reallocates the memory (https://www.codesdope.com/c-dynamic-memory/) |

double atof(const char *str)

Converts the string *str* to a decimal. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped and the conversion will start from the character '1'. The conversion stops if there is a character which is not a number. For example, in the string "123a12", the conversion will stop after converting "123" as soon as it encounters the non-numeric character 'a'.

If the string *str* doesn't form any valid floating-point number then 0.0 is returned.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[10] = "3.14";
    printf("The value is %f\n",atof(str));
    return 0;
}
```

Output

The value is 3.140000






int atoi(const char *str)

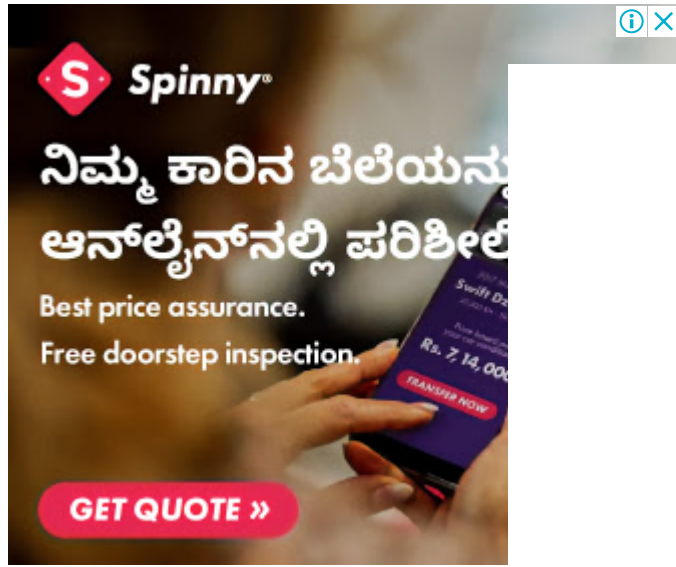
Converts the string *str* to an integer. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped and the conversion will start from the character '1'. The conversion stops if there is a character which is not a number. For example, int the string, "123a12" the conversion will stop after converting "123" as soon as it encounters the non-numeric character 'a'.

If the string *str* doesn't form any valid integer number then 0 is returned.



MOST POPULAR

- [C++ : Linked lists in C++ \(Singly linked list\) \(/blog/article/c-linked-lists-in-c-singly-linked-list/\)](/blog/article/c-linked-lists-in-c-singly-linked-list/)
May 30, 2017
- [12 Creative CSS and JavaScript Text Typing Animations \(/blog/article/12-creative-css-and-javascript-text-typing-animati/\)](/blog/article/12-creative-css-and-javascript-text-typing-animati/)
Nov. 11, 2018
- [Inserting a new node to a linked list in C++ \(/blog/article/inserting-a-new-node-to-a-linked-list-in-c/\)](/blog/article/inserting-a-new-node-to-a-linked-list-in-c/)
May 30, 2017
- [Inserting a new node in a linked list in C. \(/blog/article/inserting-a-new-node-in-a-linked-list-in-c/\)](/blog/article/inserting-a-new-node-in-a-linked-list-in-c/)
May 25, 2017
- [Set, toggle and clear a bit in C \(/blog/article/set-toggle-and-clear-a-bit-in-c/\)](/blog/article/set-toggle-and-clear-a-bit-in-c/)
July 10, 2018

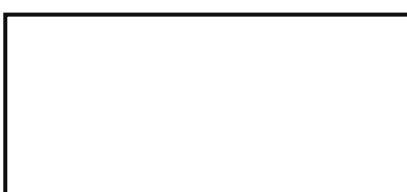

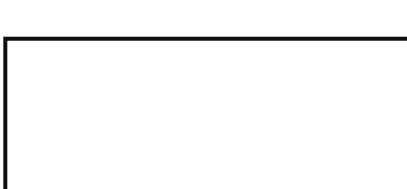
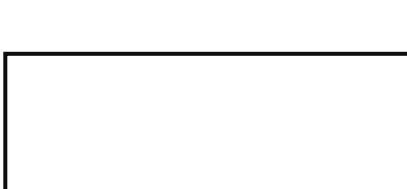
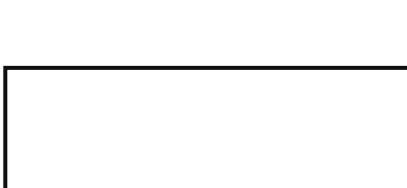


FOLLOW US

(<https://www.facebook.com/codesdope/>) (<https://www.youtube.com/codesdope/>) (<https://twitter.com/codesdope/>) (<https://www.instagram.com/codesdope/>)

(<https://www.linkedin.com/company/codesdope/>)

RECENT

- [pow\(\) in Python \(/blog/article/pow-in-python/\)](/blog/article/pow-in-python/)
April 5, 2021
- [Dutch National Flag problem - Sort 0, 1, 2 in an array \(/blog/article/dutch-national-flag-algorithm/\)](/blog/article/dutch-national-flag-algorithm/)
March 30, 2021
- [memoryview\(\) in Python \(/blog/article/memoryview-in-python/\)](/blog/article/memoryview-in-python/)
March 27, 2021
- [next\(\) in Python \(/blog/article/next-in-python/\)](/blog/article/next-in-python/)
March 26, 2021
- [map\(\) in Python \(/blog/article/map-in-python/\)](/blog/article/map-in-python/)
March 25, 2021

🔍 | Business Focus

You Might Also Like

Mouse Rollover Zoom Effect on Images

(/blog/article/mouse-rollover-zoom-effect-on-images/)

Important functions in math.h library of C

(/blog/article/important-functions-in-math-h-library-of-c/)

Formatting the print using printf in C

(/blog/article/formatting-the-print-using-printf-in-c/)


```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[10] = "55";
    printf("The value is %d\n",atoi(str));
    return 0;
}
```

Output

The value is 55

long int atol(const char *str)

Converts the string *str* to a long int. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped and the conversion will start from the character '1'. The conversion stops if there is a character which is not a number. For example, int the string, "123a12" the conversion will stop after converting "123" as soon as it encounters the non-numeric character 'a'.

If the string *str* doesn't form any valid long int then 0 is returned.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[10] = "55345323";
    printf("The value is %ld\n",atol(str));
    return 0;
}
```

Output

The value is 55345323

long long int atoll(const char *str)

Converts the string *str* to a long long int. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped and the conversion will start from the character '1'. The conversion stops if there is a character which is not a number. For example, int the string, "123a12" the conversion will stop after converting "123" as soon as it encounters the non-numeric character 'a'.

If the string *str* doesn't form any valid long long int then 0 is returned.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[20] = "55345323218721";
    printf("The value is %Ld\n",atoll(str));
    return 0;
}
```

Output

The value is 55345323218721

double strtod(const char *str, char **endptr)

Converts the string *str* to a double. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped and the conversion will start from the character '1'. The conversion stops if there is a character which is not a number. For example, int the string, "123a12" the conversion will stop after converting "123" as soon as it encounters the non-numeric character 'a'.

If the string *str* doesn't form any valid floating-point number then zero is returned.

endptr - endptr points to the first character which was not converted by the function. It just indicates where the conversion was stopped. It works if the pointer passed to the function was not a null pointer(NULL).

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[20] = "123 Random string";
    char *endptr;
    printf("The value is %lf\n",strtod(str, &endptr));
    printf("String is %s\n",endptr);
    return 0;
}
```

Output

The value is 123.000000

String is Random string

float strttof(const char *str, char **endptr)

Converts the string *str* to a float. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped and the conversion will start from the character '1'. The conversion stops if there is a character which is not a number. For example, int the string, "123a12" the conversion will stop after converting "123" as soon as it encounters the non-numeric character 'a'.

If the string *str* doesn't form any valid floating-point number then zero is returned.

endptr - endptr points to the first character which was not converted by the function. It just indicates where the conversion was stopped. It works if the pointer passed to the function was not a null pointer(NULL).

long int strtol(const char *str, char **endptr, int base)

Converts the string *str* to a long integer. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped.

endptr - endptr points to the first character which was not converted by the function. It just indicates where the conversion was stopped. It works if the pointer passed to the function was not a null pointer(NULL).

base - It is the base or radix of the number(for example, binary system has base 2, decimal number system has base 10, etc.) which has its value between 2 and 36. If the passed integer is 0 then the base is determined by the format in of the string *str*.

If the string *str* doesn't form any valid long integer number then zero is returned.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[30] = "123 1000 186A0";
    char *endptr;
    Long int a,b,c;
    a = strtol(str, &endptr, 10);
    b = strtol(endptr, &endptr, 2);
    c = strtol(endptr, &endptr, 16);
    printf("The values are %ld, %ld and %ld\n",a,b,c);
    return 0;
}
```

Linked list traversal using loop and recursion in c++

(/blog/article/linked-list-traversal-using-loop-and-recursion-in-)

Calculator using Java Swing and AWT with source code

(/blog/article/calculator-using-java-swing-and-awt-with-source-co/)

Animate your Website Elements with CSS Transforms

(/blog/article/animate-your-website-elements-with-css-transforms/)

Controlling the Outline Position with outline-offset

(/blog/article/controlling-the-outline-position-with-outline-offs/)

Prime numbers using Sieve Algorithm in C

(/blog/article/prime-numbers-using-sieve-algorithm-in-c/)

Output

The values are 123, 8 and 100000

long double strtold(const char *str, char **endptr)

Converts the string *str* to a long double. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped.

endptr - endptr points to the first character which was not converted by the function. It just indicates where the conversion was stopped. It works if the pointer passed to the function was not a null pointer(NULL).

If the string *str* doesn't form any valid long double number then zero is returned.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[30] = "1234.56";
    printf("The value is %lf\n",strtold(str, NULL));
    return 0;
}
```

Output

The value is 1234.560000

long long int strtoll(const char *str, char **endptr, int base)

Converts the string *str* to a long long integer. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped.

endptr - endptr points to the first character which was not converted by the function. It just indicates where the conversion was stopped. It works if the pointer passed to the function was not a null pointer(NULL).

base - It is the base or radix of the number(for example, binary system has base 2, decimal number system has base 10, etc.) which has its value between 2 and 36. If the passed integer is 0 then the base is determined by the format in of the string *str*.

If the string *str* doesn't form any valid long integer number then zero is returned.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[30] = "123456789";
    printf("The value is %lld\n",strtoll(str, NULL, 10));
    return 0;
}
```

Output

The value is 123456789

unsigned long int strtoul(const char *str, char **endptr, int base)

Converts the string *str* to an unsigned long integer. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped.

endptr - endptr points to the first character which was not converted by the function. It just indicates where the conversion was stopped. It works if the pointer passed to the function was not a null pointer(NULL).

base - It is the base or radix of the number(for example, binary system has base 2, decimal number system has base 10, etc.) which has its value between 2 and 36. If the passed integer is 0 then the base is determined by the format in of the string *str*.

If the string *str* doesn't form any valid long integer number then zero is returned.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[30] = "123456789";
    printf("The value is %lu\n",strtoul(str, NULL, 10));
    return 0;
}
```

Output

The value is 123456789

unsigned long long int strtoull(const char *str, char **endptr, int base)

Converts the string *str* to a unsigned long long integer. If there are any white-space characters at the beginning of the string *str* then they are skipped. For example, in the string " 123", white-space characters present at the beginning of the string will be skipped.

endptr - endptr points to the first character which was not converted by the function. It just indicates where the conversion was stopped. It works if the pointer passed to the function was not a null pointer(NULL).

base - It is the base or radix of the number(for example, binary system has base 2, decimal number system has base 10, etc.) which has its value between 2 and 36. If the passed integer is 0 then the base is determined by the format in of the string *str*.

If the string *str* doesn't form any valid long integer number, then zero is returned.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char str[30] = "123456789";
    printf("The value is %llu\n",strtoull(str, NULL, 10));
    return 0;
}
```

Output

The value is 123456789

int rand(void)

Returns a pusedo-random integral number. The range is in between 0 and RAND_MAX(inclusive).

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    int i;
    for(i=1;i<=10;i++)
    {
        printf("%d\n",rand());
    }
    return 0;
}
```

Output

1804289383
846930886
1681692777
1714636915
1957747793
424238335
719885386
1649760492
596516649
1189641421

void srand(unsigned int seed)

srand seeds the pseudo-random number generator used by the rand function. The pseudo-random number generator is expected to produce a different sequence of random numbers for different values of *seed*.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main()
{
    int i;
    srand(time(NULL));
    for(i=0;i<10;i++)
    {
        printf("%d\n",rand());
    }
    return 0;
}
```

Output

1804289383
1585335290
1205554746
1968078301
590011675
290852541
1045618677
757547896
444454915
1215069295

void *bsearch(const void *key, const void *base, size_t n, size_t size, int (*compar)(const void *, const void *))

Performs binary search on an array pointed by *base*.

key - key for the search.
base - pointer to the array where the search is to be performed.
n - Number of elements in the array pointed by *base*.
size - size of elements of the array in bytes.
compar - Function to compare two elements. It should compare the key element and the current element and return a negative value if key element comes before the current element, positive is it is after the current element and o if it is equivalent. This function is made in the example given below.

```
#include <stdio.h>
#include <stdlib.h>

//compare function
int cmp(const void *a, const void *b)
{
    return ( *(int*)a - *(int*)b );
}

int a[] = {34,45,23,56,12};

int main()
{
    int key = 23;
    int *i;

    i = (int*) bsearch (&key, a, 5, sizeof(int), cmp);
    if(i!=NULL)
        printf("Found\n");
    else
        printf("Not found\n");
    return 0;
}
```

Output

Found

void qsort(void *base, size_t n, size_t size, int (*compar)(const void*, const void*))

Performs quicksort on the array pointed by base.*base* - pointer to the array.
n - Number of elements in the array pointed by base.
size - size of elements of the array in bytes.
compar - Function to compare two elements. It should compare the key element and the current element and return a negative value if key element comes before the current element, positive is it is after the current element and o if it is equivalent. This function is made in the example given below.

```
#include <stdio.h>
#include <stdlib.h>

//compare function
int cmp(const void * a, const void * b)
{
    return ( *(int*)a - *(int*)b );
}

int a[] = {34,45,23,56,12};

int main()
{
    int i;

    qsort(a, 5, sizeof(int), cmp);
    for(i=0;i<5;i++)
        printf("%d\n",a[i]);
    return 0;
}
```

Output

12
23
34
45
56

int abs(int n)

Returns absolute value of integer n.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("%d\n",abs(-1));
}
```

Output

1

long int labs(long int n)

Returns absolute value of long integer n.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("%lld\n", labs(-12345678912));
    return 0;
}
```

Output

12345678912

long long int llabs(long long int)

Returns absolute value of long long integer n.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    printf("%lld\n", llabs(-1234567891212344));
    return 0;
}
```

Output

123456789121234

div_t div(int numer, int denom)

div takes two arguments - the numerator and the denominator for the division and returns a structure *div_t* with two members - quotient(quot) and remainder(rem).

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    div_t result;
    result = div(5,3);
    printf("The result of division of 5 and 3\n%d is quotient and %d is the remainder\n", result.quot, result.rem);
    return 0;
}
```

Output

The result of division of 5 and 3
1 is quotient and 2 is the remainder

ldiv_t ldiv(long int numer, int denom)

ldiv takes two arguments - the numerator and the denominator for the division and returns a structure *ldiv_t* with two members - quotient(quot) and remainder(rem).

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    ldiv_t result;
    result = ldiv(5512458745155, 3541525);
    printf("The result of division is\n%d is quotient and %ld is the remainder\n", result.quot, result.rem);
    return 0;
}
```

Output

The result of division is
155 is quotient and 2308780 is the remainder

lldiv_t lldiv(long long int numer, long long int denom)

lldiv takes two arguments - the numerator and the denominator for the division and returns a structure *lldiv_t* with two members - quotient(quot) and remainder(rem).

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    lldiv_t result;
    result = lldiv(5512458745155, 354254771525);
    printf("The result of division is\n%lld is quotient and %lld is the remainder\n", result.quot, result.rem);
    return 0;
}
```

Output

The result of division is
15 is quotient and 198637172280 is the remainder

void abort(void)

abort causes the termination of the program.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    printf("Hello\n");
    abort();
    printf("How are you?\n");
    return 0;
}
```

Output

Hello
Aborted (core dumped)

int atexit (void(*func)(void))

atexit is used to call a function at the normal termination of the program. The function pointed by *func* automatically gets called at the termination. If more than one functions are specified then the execution occurs in a reverse order, i.e., the last function is executed first.

```
#include <stdio.h>
#include <stdlib.h>

void f1(void)
{
    printf("Function 1\n");
}

void f2(void)
{
    printf("Function 2\n");
}

int main ()
{
    printf("Hello\n");
    atexit(f1);
    atexit(f2);
    return 0;
}
```

Output

Hello
Function 2
Function 1

int at_quick_exit(void(*func)(void))

It is executed when *quick_exit* is called.

Return value - 0 in success, non-zero in the case of failure.

```
#include <stdio.h>
#include <stdlib.h>

void f1(void)
{
    printf("Function 1\n");
}

void f2(void)
{
    printf("Function 2\n");
}

int main ()
{
    printf("Hello\n");
    at_quick_exit(f1);
    at_quick_exit(f2);
    quick_exit(0);
    printf("After quick exit\n");
    return 0;
}
```

Output

Hello
Function 2
Function 1

_Noreturn void quick_exit(int status)

_Noreturn - Specifies that the function does not return to where it was called from.

Terminates a program normally after calling the functions registered using *at_quick_exit*. Additional clean-ups are not performed at *quick_exit*.

status
zero or *EXIT_SUCCESS* - Successful termination status is returned to the host environment.
EXIT_FAILURE - Unsuccessful termination status is returned to the host environment.

```
#include <stdio.h>
#include <stdlib.h>

void f1(void)
{
    printf("Function 1\n");
}

void f2(void)
{
    printf("Function 2\n");
}

int main ()
{
    printf("Hello\n");
    at_quick_exit(f1);
    at_quick_exit(f2);
    quick_exit(0);
    printf("After quick exit\n");
    return 0;
}
```

Output

Hello
Function 2
Function 1

void exit (int status)

exit is used to terminate a program normally after performing the regular clean-ups.

status
zero or *EXIT_SUCCESS* - Successful termination status is returned to the host environment.
EXIT_FAILURE - Unsuccessful termination status is returned to the host environment.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    printf("Hello\n");
    exit(0);
    printf("After quick exit\n");
    return 0;
}
```

Output

Hello

char *getenv(const char *name)

Returns the enviornment for the operating system (pointer to a string) associated with the *name*.

name
Environment to find in the operating system.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    char* p;
    p = getenv("PATH");
    printf("%s\n",p);
    return 0;
}
```

int system(const char *command)

system is used to run another program.

command - Command to be executed.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    system("ls");
    return 0;
}
```

void _Exit(int status)

Similar to *exit* but doesn't performs the clean-up task.

```
#include <stdio.h>
#include <stdlib.h>

int main ()
{
    printf("Hello\n");
    _Exit(0);
    printf("After quick exit\n");
    return 0;
}
```

Output

Hello

You can learn about the functions `calloc` (<https://www.codesdope.com/c-dynamic-memory/>), `malloc` (<https://www.codesdope.com/c-dynamic-memory/mallocm>) (<https://www.codesdope.com/c-dynamic-memory/>) `realloc` (<https://www.codesdope.com/c-dynamic-memory/realloc>) (<https://www.codesdope.com/c-dynamic-memory/>) and free in detail from the Dynamic memory chapter of the C course (<https://www.codesdope.com/c-dynamic-memory/>).

How to Create a Digital Clock using JavaScript

10 Amazing Effects You Can Create Using Box Shadows

Properties in C# - Static Property, Automatic Property and Property...

Delegate in C# - Passing methods to methods.

Merge Sort and its analysis


Creating Different Box Shadow Effects using CSS

Getting Notebook Paper Effect with CSS


Find elerr unsc




Liked the post?

[\(https://www.facebook.com/sharer/sharer.php?u=https://www.codesdope.com/blog/article/important-functions-in-the-stdlibh-library-of-c/\)](https://www.facebook.com/sharer/sharer.php?u=https://www.codesdope.com/blog/article/important-functions-in-the-stdlibh-library-of-c/)




[functions-in-the-stdlibh-library-of-c\)](https://twitter.com/intent/tweet?url=https://www.codesdope.com/blog/article/important-functions-in-the-stdlibh-library-of-c&text=important%20functions%20in%20the%20stdlib.h%20library%20of%20C&via=codesdope)

[in the stdlib.h library of C &via=codesdope\)](https://www.linkedin.com/shareArticle?url=https://www.codesdope.com/blog/article/important-functions-in-the-stdlibh-library-of-c&title=important%20functions%20in%20the%20stdlib.h%20library%20of%20C)

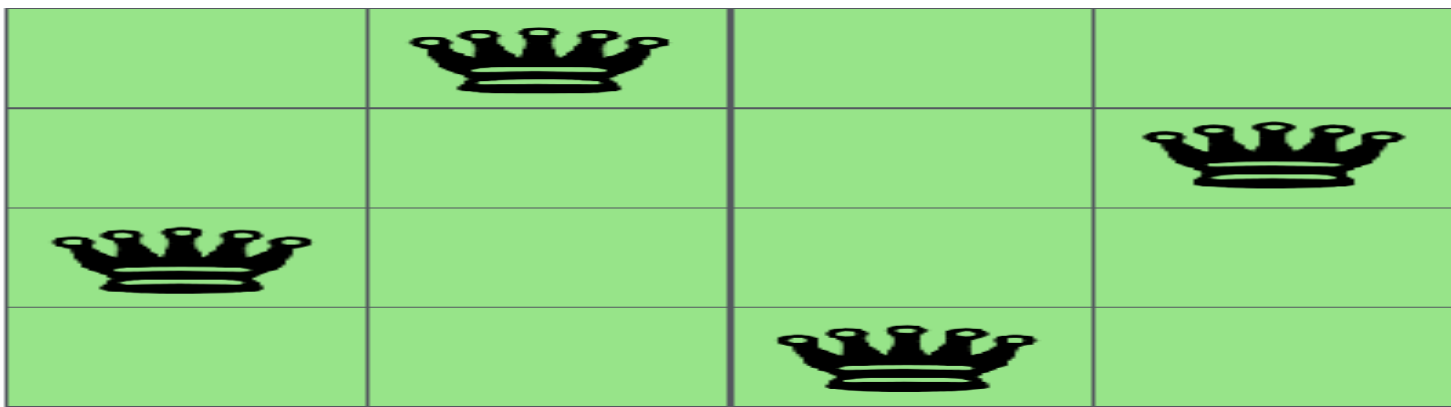
[in the stdlib.h library of C\)](https://pinterest.com/pin/create/bookmarklet/?media=https://www.codesdope.comhttps://codesdope-media.nyc3.cdn.digitaloceanspaces.com/prod/media/blog_images/110/blog_cover_pica/article_110_small.png&url=https://www.codesdope.com/blog/article/important-functions-in-the-stdlibh-library-of-c&description=important%20functions%20in%20the%20stdlib.h%20library%20of%20C)

Amit Kumar (/blog/author/54322/?author=54322)

Developer and founder of CodesDope.

 [/](https://www.facebook.com/codesdope)  [/](https://www.twitter.com/codesdope)  [/](https://www.linkedin.com/in/amit-kumar-66903395)

Editor's Picks




0 COMMENT

Please login (/accounts/login/?next=/blog/article/important-functions-in-the-stdlibh-library-of-c/) to view or add comment(s).





CATEGORIES

- [PRO \(/pro.codesdope.com/\)](https://pro.codesdope.com/)
- [ABOUT \(/about/\)](#)
- [COURSES \(/\)](#)
- [TERMS AND CONDITIONS \(/terms-of-use/\)](#)
- [PRIVACY POLICY \(/privacy-policy/\)](#)
- [CONTACT US \(/contact-us/\)](#)
- [ADVERTISEMENT \(/advertise-with-us/\)](#)

KEEP IN TOUCH

 [help@codesdope.com \(/mailto:help@codesdope.com\)](mailto:help@codesdope.com)

FOLLOW US

 [/](https://www.facebook.com/codesdope)  [/](https://twitter.com/codesdope)  [/](https://www.linkedin.com/company/codesdope)  [/](https://www.pinterest.com/codesdope/)