I'm Confused! Char, Char* or String in C++

Ask Question

Asked 9 years, 9 months ago Modified 8 years, 10 months ago Viewed 4k times

5

when i use char or char*, visual studio 2012(11) only couts last character such as:

```
#include <iostream>
#include <string>
int main(){
 using namespace std;
 char chName = 'Alex';
 cout<<chName;
}
```

It displays only "x". it is correct is i use `string strName = "Alex"` but in those function which have parameter as a char, string can not be passed on as argument. in this case, VS compiler says that strings cant be converted into int. also tell me what is difference between char and char*. I am a PHP developer and C++ is so confusing. Please help me.

c++ c-strings

Share Edit Follow Flag

asked Jun 19, 2012 at 16:44
Suchal
193 ●1 ●7

4 When using C++ use string. Inside double quotes is a constant string.If you want to return a char* from a string use **c_str()**. Check the link stackoverflow.com/questions/388242/... – DumbCoder Jun 19, 2012 at 16:48

Add a comment

Start a bounty

4 Answers

Active Oldest Votes

8

`char` can only keep 1 character at a time; in this case, it keeps your last character, `'x'`. `char *` is a pointer to one or more `char` objects, and if read correctly, can also be used as a string. So setting

```
const char *chName = "Alex";
cout << chName;
```

should output the whole name.

Another problem is your use of quotations. `'x'` denotes a `char`, while `"x"` denotes a array of `char` s, known as a string literal.

If there is a function that requires you to pass a char *, you could pass

```
const char *param = "this is da parameter";
function (param);
```

or

```
std::string param = "da parameter"; //std::string is a type of string as well.
function (param.c_str ());
```

You could also use the declaration

```
char chName[] = "Alex";
```

This would create an local array of `char` s (namely, 5 `char` s, because it appends a null character at the end of the array). So, calling `chName[3]` should return `'x'`. This can also be streamed to `cout` like the others

```
cout << chName;
```

EDIT: By the way, you should return an `int` in your function `main ()`. Like `0`.

Share Edit Follow Flag

edited Jun 19, 2012 at 17:35

answered Jun 19, 2012 at 16:49
GILGAMESH
1,687 ●3 ●20 ●32

2 I don't know about "you should return an int in your function main." C++ allows main (but no other function) to omit the return statement. If it is omitted, 0 is returned. – Rob$_\varphi$ Jun 19, 2012 at 17:10

1 @Rob$_\varphi$: C++ allows a lot of silly things. You should return an `int` from `main` even if you don't *have* to. – Mooing Duck Jun 19, 2012 at 17:26

@MooingDuck, what for, if `return 0` is implicit? – Griwes Jun 19, 2012 at 17:48

@MooingDuck, both of them are nonsense in this case. – Griwes Jun 19, 2012 at 19:39

*" If there is a function that requires you to pass a char *, you could pass "* No nonono, only if it's one that wants a `const char*`. – Hatted Rooster Nov 29, 2016 at 9:48

Add a comment

4

```
char chName = 'Alex';
```

This is wrong. This way, you create a four-byte integer out of Alex, then you store it in a char -- of course, it doesn't fit into a one-byte char, so only its less significant byte, the `x` gets stored, which is then output. You need to use

```
const char *chName = "Alex";
```

to get the correct output.

An alternative to your problem: use `std::string`, as you're working in C++, but for those functions who expect a `char *` as their argument, use:

```
std::string str; // C++ string object
function_call(str.c_str());
```

Share Edit Follow Flag

edited May 21, 2013 at 11:40
Ken D
5,700 ●1 ●33 ●58

answered Jun 19, 2012 at 16:47
user529758

4 Better to use a proper C++ `string` though, rather than an *old skool* C-style `char *`. – Paul R Jun 19, 2012 at 16:48

I'm not saying which is better; I'm answering OP's question. (Btw C being "old skool" is not necessarily true. Try to write an operating system in C...) – user529758 Jun 19, 2012 at 16:50

@H2CO3: Writing an operating system is a completely niche use and does not whatsoever prevent C from being a worthless pile of junk in the general case. – Puppy Jun 19, 2012 at 16:51

1 @H2CO3: We're talking about C++, not C. `char*` is definitely outmoded for most purposes in C++. – Fred Larson Jun 19, 2012 at 16:53

"prevent C from being a worthless pile of junk in the general case" Have you ever written a single line of C? – user529758 Jun 19, 2012 at 16:53

Add a comment | Show **2** more comments

3

The `char` type holds a single integral value, usually with the range -128 to 127. It is not a string type. Single quotes in C are used for character literals, not string literals. `'Alex'` is not the same as `"Alex"`.

The char literal syntax you used:

```
char chName = 'Alex';
```

is called a multi-character literal, and it has an implementation defined value of type `int`. Implementations I'm familiar with construct this by concatenating the values of the individual characters. So the value of `'Alex'` is probably A 0x41, l 0x6C, e 0x65, x 0x78, or `0x416C6578`. Then when you assign it to a char it gets truncated to just the last byte (since thats all a char can hold), which is 0x78, or the same as `'x'`.

The `*` is the pointer deference operator in C, and it's also used when declaring a pointer. So `const char *chName = "Alex";` declares a pointer to a `char` rather than a single `char`, and that pointer will point to the first character of the string literal `"Alex"`. So your program might look like:

```
#include <iostream>

int main(){
  const char *chName = "Alex";
  std::cout << chName;
}
```

Share Edit Follow Flag

edited Jun 19, 2012 at 17:27

answered Jun 19, 2012 at 16:50
bames53
82.4k ●13 ●167 ●234

Add a comment

2

```
char chName = 'Alex';
```

is a multicharacter literal and it is implementation defined.

```
_ C++ standard, §2.14.3/1 - Character literals _

An ordinary character literal that contains more than one c-char is a multicharac
literal . A multicharacter literal has type int and implementation-defined value.
```

instead of this, you should use

```
const char *chPTR = "Alex";
```

or

```
char chName[] = "Alex";
```

Difference between char and char*
In `char ch;` ch is a *char variable* which can store a single ascii character, whereas `char *ch;` is a *pointer to char* which can store address of a char variable.

Difference between char and String
see this SO post.

Share Edit Follow Flag

edited May 23, 2017 at 10:24
Community Bot
1 ●1

answered Jun 19, 2012 at 16:52
Eight
4,096 ●5 ●27 ●51

Add a comment

Your Answer

Community wiki

Post Your Answer

Not the answer you're looking for? Browse other questions tagged c++ c-strings or ask your own question.

Linked
4235 The Definitive C++ Book Guide and List
145 Difference between string and char[] types in C++

Related
3241 How do I iterate over the words of a string?
1013 How to convert a std::string to const char* or char*
77 error C2065: 'cout' : undeclared identifier
1 Cannot assign pointer in a self-referential object in Visual Studio 2010
1921 Easiest way to convert int to string in C++
433 std::string to char*
12 c++11 sorting list using lambda
1 Invalid operands to binary expression when using unordered_map?