[Skip to main content](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/courseware/a71234edfd4f465e8d3e7e0082d04540/fa25de4817d044fea3b65cf24587a31b/3?activate_block_id=block-v1%3AIITBombayX%2BCS101.2x%2B2022_T1%2Btype%40vertical%2Bblock%404c98fb4b34b646f6a84c7cbc99afc968" \l "main)

**[IITBombayX Home Page](http://iitbombayx.in/)**

**IITBombayX: CS101.2xObject-Oriented Programming**

[Courses](http://iitbombayx.in/courses)

[Help](https://iitbombayx.in/contact_us)

[[https://courses.iitbombayx.in/static/images/profiles/default_50.3455a6581573.png](https://courses.iitbombayx.in/dashboard) Dashboard for:Paramhans\_Singh](https://courses.iitbombayx.in/dashboard)

1. [Course, current location](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/course/)

1. [Discussion](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/discussion/forum/)

1. [Progress](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/progress)

1. [Grading Policy](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/487375f5363a49a492641b63d5e814ec/)

1. [Course Slides](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/pdfbook/0/)

1. [Practice Programming](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/b5c2b3240ef9416ebb0ea5b2087cb4ee/)

1. [FAQs](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/f2e9c32b35ae4ef083ed716b4981ff0d/)

[Course](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/course/)  [Topic 2: Review of Pointers and Structures](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/course/#block-v1:IITBombayX+CS101.2x+2022_T1+type@chapter+block@a71234edfd4f465e8d3e7e0082d04540)  [Session 3: Use of Pointers in C++ Programs](https://courses.iitbombayx.in/courses/course-v1:IITBombayX+CS101.2x+2022_T1/course/#block-v1:IITBombayX+CS101.2x+2022_T1+type@sequential+block@fa25de4817d044fea3b65cf24587a31b)  Walkthrough

 PreviousNext

1. video Use of Pointers in C++ Programs
2. other Slides: Use of Pointers in C++ Programs
3. other Walkthrough
4. other Walkthrough

**Walkthrough**

 Bookmark this page

**Behaviour of Character Pointers**

**objective:**

Till now, you have got some basic understandings of how a typical pointer behaves. But when you start observing the behaviour of character pointers, things don't work you expect them to work like. Character pointers work a lot different than what other pointers do and it's essential to be aware of their anomalous behaviour.

Not only the pointers, the character arrays also don't behave like the arrays of other datatypes. Let's first start with them:

**Behaviour of the character arrays:**

#include <iostream>

using namespace std;

int main() {

       char charr[] = "TEXT";

                cout<<charr;

}

What output do you expect ? Had it been an integer array then the output would have been the memory address of the first element of the array. (As you know that the array is the pointer to its first element). But here something different happens.

**Output:**TEXT

As you can see, the character array behaves much differently than the array corresponding to other data types. But can you say here that the character array points to its first element i.e. Can you say that charr points to the address of the character 'T'?

Let's see! What will be the output of  cout<<\*(charr); ?

The output is T.

Thus, we can say that the array still points to its first element but yet behaves differently compared to other data type arrays.

**Behaviour of character pointers:**

#include <iostream>

using namespace std;

int main()

{

       char charr[] = "TEXT";

       char\* ch = charr;

                cout<<ch<<" "<<\*ch;

 }

What output do we expect for this? Had it been a normal integer pointer then cout<<ch would have given the address of the first element of array charr and cout<<\*ch would have given the value of the first element. Well, you might have got the idea by now that the output is not going to be as expected here.

**Output:**TEXT T

So, the second output is as expected but the output of cout<<ch is not what we thought it would be.

Let's see one more interesting thing.

What should be the output of cout<<&(\*ch) ? It's surprising to notice that it's still giving the output "TEXT".

**Why this behaviour ?**

So basic reason why this behaviour is observed is because the cout operator is overloaded for const void \* and const char \*. The pointers of int are converted to const void \*  while the char \*  pointers are kept as const char \*. The cout operator is overloaded such that it outputs the address pointed to by the pointer when it gets  const void \* while it treats  const char \*  as a C-string and prints the characters till it gets a null character.

**What if we want to know the address pointed to by the character pointer?**

The way we can do this is by explicitly type-casting the char\* pointer to void\* before or during passing it to cout operator. This can be done as follows:

cout << static\_cast<const void\*>(ch) << endl;

This makes the cout operator print the address pointed to by the ch.

**Conclusion:**

The conclusion from the above observations are that you should keep these things in mind while dealing with the character pointers and try to explore more about the behaviour of these pointers through small experiments as shown above.

 PreviousNext

© All Rights Reserved

[IITBombayX Home Page](http://iitbombayx.in/)

1. [About](http://iitbombayx.in/about)
2. [Research](http://iitbombayx.in/research)
3. [FAQs](http://iitbombayx.in/faq)
4. [Contact Us](http://iitbombayx.in/contact_us)

* [MOOCs Copyright](http://iitbombayx.in/mooc-copyright-guidelines)
* [Terms of Service and Honour Code](http://iitbombayx.in/honor)
* [Authorization Document](http://iitbombayx.in/authorization)
* [Privacy Policy](http://iitbombayx.in/privacy)

[Powered by Drupal](https://www.drupal.org/)

[Creative Common License](https://creativecommons.org/)

[](http://open.edx.org/)