مبانی کامپیوتر و برنامه سازی

دانشگاه آل طه

رشته کامپیوتر

درس تخصصی

پاییز 1403

رئوس مطالب

- مفاهیم اولیه
- محاسبات در کامپیوتر
- آشنایی با الگوریتم و فلوچارت
- مقدمات برنامه سازی
- فرمت بندی ورودی/خروجی
- دستورها
- توابع
- آزمون و خطایابی برنامه

- آرایه ها
- اشاره گرها
- کاراکترها و رشته ها
- ساختارها
- آشنایی با c++
- آشنایی با رده ها
- حوزه های private , public
- پرونده ها

جلسه چهارم

- تاریخچه مختصر زبان برنامه‌نویسی C++
- در اوایل دهه 1980، اولین نسخه‌های C++با نام " Cبا کلاس‌ها" منتشر شدند. به تدریج ویژگی‌های جدیدی به زبان اضافه شد و در نهایت در سال 1998، اولین استاندارد رسمی C++منتشر شد. از آن زمان تاکنون، C++به طور مداوم در حال توسعه و بهبود است.

سال	استاندارد C++	نام غیررسمی
۱۹۹۸	ISO/IEC 14882:1998	C++98
۲۰۰۳	ISO/IEC 14882:2003	C++03
۲۰۱۱	ISO/IEC 14882:2011	C++11
۲۰۱۴	[۱۳]ISO/IEC 14882:2014	C++14
۲۰۱۷	[۱۴]ISO/IEC 14882:2017	C++17, C++1z
۲۰۲۰	[۱۵]ISO/IEC 14882:2020	C++20, C++2a
۲۰۲۳		C++23, C++2b

چرا ++C؟

- برنامه‌نویسی شیءگرا
- کارایی بالا
- انعطاف‌پذیری
- سطح بالا و پایین

- توسعه سیستم‌عامل‌ها (لینوکس، ویندوز)
- توسعه بازی‌های ویدیویی
- برنامه‌نویسی شبکه
- برنامه‌نویسی علمی و مهندسی
- برنامه‌نویسی پایگاه داده

تفکر شیء گرایی (Object-Oriented Programming یا OOP)

- شیء گرایی: رویکردی نوین در برنامه‌نویسی
- دنیای واقعی را تصور کنید. همه چیز اطراف ما از اشیاء تشکیل شده است. هر شیء دارای ویژگی‌ها (مثلاً رنگ، اندازه) و رفتارهایی (مثلاً حرکت کردن، صدا کردن) است.
- شیء گرایی به ما این امکان را می‌دهد تا این مفاهیم را در برنامه‌نویسی پیاده‌سازی کنیم. در برنامه‌نویسی شیء گرا، برنامه‌ها به عنوان مجموعه‌ای از اشیاء در نظر گرفته می‌شوند که با یکدیگر تعامل دارند.

آشنایی با مفاهیم اصلی

کلاس :

- تعریف :یک قالب یا طرح برای ایجاد اشیاء است .
- مثال :کلاس "ماشین" می‌تواند ویژگی‌هایی مانند رنگ، مدل و سرعت و رفتارهایی مانند حرکت کردن و ترمز کردن داشته باشد .

شیء :

- تعریف :نمونه‌ای از یک کلاس است .
- مثال :یک ماشین خاص با رنگ قرمز و مدل پژو یک شیء از کلاس "ماشین" است .

آشنایی با مفاهیم اصلی

ویژگی :

- تعریف :مشخصه یا داده‌ای که به یک شیء تعلق دارد .
- مثال :رنگ، مدل و سرعت ویژگی‌های یک ماشین هستند .

رفتار :

- تعریف :عملی که یک شیء می‌تواند انجام دهد .
- مثال :حرکت کردن، ترمز کردن و شتاب گرفتن رفتارهای یک ماشین هستند .

وراثت :

- تعریف :فرآیندی که در آن یک کلاس (کلاس فرزند) ویژگی‌ها و رفتارهای کلاس دیگری (کلاس والد) را به ارث می‌برد .
- مثال :کلاس "ماشین مسابقه‌ای" می‌تواند از کلاس "ماشین" ارث‌بری کند و ویژگی‌ها و رفتارهای جدیدی مانند سرعت بیشتر را به آن اضافه کند.

مفهوم کلاس در c++

- کلاس در زبان برنامه‌نویسی C++، یک طرح یا الگویی است که مشخص می‌کند یک شیء چه ویژگی‌ها attributes و رفتارها methodsخواهد داشت.
- به عبارت ساده‌تر، کلاس یک نقشه ساختمانی است که تعیین می‌کند یک شیء چگونه ساخته شود و چه قابلیت‌هایی داشته باشد.
- کلاس شامل اشیایی با خواص مشترک می شود.

دستور العمل های برنامه c++دارای چه ویژگی هایی هستند؟

- حساسیت به حروف بزرگ و کوچک
  - هر کلمه کلیدی، متغیر، تابع یا کلاس در c++باید دقیقاً با همان حروف بزرگ و کوچک نوشته شود. مثلاً int با Intمتفاوت است.
- نقطه ویرگول (;)
  - هر دستور در c++باید با یک نقطه ویرگول به پایان برسد.
- هر دستور میتواند در یک سطر یا چند سطر ادامه داشته باشد
- در هر سطر میتوان چند دستور تایپ کرد(توصیه نمیشود)
- توضیحات در بین /* و */ قرار میگیند. یا بعد از //

انواع داده های اولیه در C++نمایش پیش‌نویس‌ها

- **انواع داده‌های عددی**
- **Int: برای ذخیره‌سازی اعداد صحیح (مثبت و منفی) استفاده می‌شود.**
- **float: برای ذخیره‌سازی اعداد اعشاری با دقت معمولی استفاده می‌شود.**
- **double: برای ذخیره‌سازی اعداد اعشاری با دقت بالا استفاده می‌شود.**
- **char: برای ذخیره‌سازی یک کاراکتر استفاده می‌شود. در واقع، کاراکترها به صورت عددی در حافظه ذخیره می‌شوند.**
- **bool: برای ذخیره‌سازی مقادیر منطقی (درست یا غلط) استفاده می‌شود.**
- **void: نشان‌دهنده عدم وجود نوع داده است و معمولاً در تعریف توابعی که مقداری برنمی‌گردانند یا اشاره گر به نوع داده‌ای نامشخص استفاده می‌شود.**

مثال

- int age = 30; // عدد صحیح
- float pi = 3.14159; // عدد اعشاری با دقت معمولی
- double e = 2.718281828459045; // عدد اعشاری با دقت بالا
- char grade = 'A'; // یک کاراکتر
- bool isStudent = true; // مقدار منطقی درست

ادامه

- signed: هم مقادیر مثبت و هم منفی را در خود نگه دارد.
- unsigned: متغیر فقط می‌تواند مقادیر غیر منفی (صفر و اعداد مثبت) را در خود نگه دارد. این حالت برای مواردی که مطمئن هستیم داده ما همیشه مثبت است، مفید است و باعث می‌شود بتوانیم از محدوده بیشتری از اعداد مثبت استفاده کنیم.
- long: متغیر بتواند اعداد بزرگتری را در خود نگه دارد.
- short: متغیر از حافظه کمتری استفاده کند، اما محدوده مقادیر قابل ذخیره‌سازی در آن کمتر است.

تعریف متغیر

- متغیرها نامی برای کلمات حافظه هستند.
- داده ها در آن قرار میگیند.
- محتوای آن ها ممکن است در طول اجرای برنامه تغییر کند

- متغیرها امکان نام گذاری برای خانه های حافظه را فراهم میکند.

- int age = 30;
- int a , b , c ;

قوانین نام‌گذاری متغیرها در C++

- حروف و اعداد :نام متغیرها می‌تواند ترکیبی از حروف بزرگ و کوچک، اعداد و زیرخط (_) باشد .
- حرف اول :نام متغیر نباید با رقم شروع شود .
- کلمات کلیدی :از کلمات کلیدی زبان C++ مانند int, float, if, else و ... به عنوان نام متغیر نمی‌توان استفاده کرد .
- حساسیت به حروف بزرگ و کوچک C++ :به حروف بزرگ و کوچک حساس است. یعنی Age وage دو متغیر متفاوت هستند .
- فضا :بین حروف نام متغیر نمی‌توان فاصله گذاشت .
- کاراکترهای خاص :از کاراکترهای خاص مانند !, @, #, $ و ... نمی‌توان در نام متغیر استفاده کرد .

جدول نحوه مقداردهی انواع داده‌ها در C++

نوع داده	روش مقداردهی اولیه	مثال	توضیحات
int	`int age = 30;`	عدد صحیح	به متغیر `age` مقدار 30 اختصاص داده می‌شود.
float	`float pi = 3.14;`	عدد اعشاری با دقت معمولی	به متغیر `pi` مقدار 3.14 اختصاص داده می‌شود.
double	`double e = 2.71828;`	عدد اعشاری با دقت بالا	به متغیر `e` مقدار 2.71828 اختصاص داده می‌شود.
char	`char grade = 'A';`	یک کاراکتر	به متغیر `grade` کاراکتر A اختصاص داده می‌شود. توجه کنید که کاراکترها درون تک کوتیشن قرار می‌گیرند.
bool	`bool isStudent = true;` یا `bool isStudent = false;`	مقدار منطقی	به متغیر `isStudent` مقدار درست (true) یا نادرست (false) اختصاص داده می‌شود.

اعلان ثوابت در c++

- ثوابت Constants به مقادیری گفته می‌شود که در طول اجرای برنامه تغییر نمی‌کنند.
- استفاده از ثوابت باعث افزایش خوانایی کد، کاهش خطاها و بهبود نگهداری کد می‌شود.
- انواع :
  1. استفاده از کلمه کلیدی :const
  2. استفاده از ماکروها ((Macrosبا #:define

استفاده از کلمه کلیدی const:

- const نوع_داده نام_ثابت = مقدار;
- const float PI = 3.14159;
- const char NEWLINE = '\n';

- در زمان کامپایل مقدار ثابت جایگزین می‌شود.
- دارای نوع داده مشخصی است و از مزایای بررسی نوع داده در زمان کامپایل بهره‌مند می‌شود.
- دامنه آن به محلی که تعریف شده است محدود می‌شود.
- در دیباگ کردن بهتر عمل می‌کند زیرا در زمان کامپایل جایگزین می‌شود.
- توصیه می‌شود از const به جای define استفاده شود.

2. استفاده از ماکروها #define

- #define نام_ثابت مقدار
- #define MAX_VALUE 100

- در زمان پیش‌پردازش مقدار ثابت جایگزین می‌شود.
- فاقد نوع داده است و ممکن است به خطاهای مربوط به نوع داده منجر شود.
- در کل فایل یا بخش‌هایی که شامل # includeمی‌شوند، قابل دسترسی است.
- در دیباگ کردن ممکن است مشکلاتی ایجاد کند.
- توصیه می‌شود از constبه جای define استفاده شود.

مزایای استفاده از ثوابت

- خوانایی کد: استفاده از نام‌های معنی‌دار برای ثوابت باعث می‌شود کد خواناتر شود.
- کاهش خطا: با استفاده از ثوابت، احتمال بروز خطا در هنگام تغییر مقادیر کاهش می‌یابد. زیرا تنها کافی است یک بار مقدار ثابت را تغییر دهیم.
- بهبود نگهداری کد: با تغییر مقدار یک ثابت، تمام قسمت‌هایی از کد که از آن ثابت استفاده می‌کنند به طور خودکار به‌روزرسانی می‌شوند.

مثال

```
#include <iostream>
using namespace std;
const double PI = 3.14159;
const int MAX_STUDENTS = 30;
int main() {
  double radius = 5.0;
  double area = PI * radius * radius;
  cout << "Area of circle: " << area << endl;
  int numStudents = 25;
  if (numStudents > MAX_STUDENTS) {
    cout << "Class is full." << endl;
  }
  return 0;
}
```

عملگرها

- عملگرها نمادهایی هستند که اعمال خاصی را نمایش میدهند. و بر روی یک یا دو عملوند(مقدار) عمل میکنند.
- انواع عملگرها
  - عملگرهای محاسباتی
  - عملگرهای رابطه ای
  - عملگرهای منطقی
  - عملگرهای ترکیبی
  - عملگرهای بیتی
  - عملگرهای متفرقه

عملگرهای محاسباتی

عملگر	نام	مثال
+	جمع	`int x = 5 + 3;` (x برابر 8 می‌شود)
-	تفریق	`int y = 10 - 4;` (y برابر 6 می‌شود)
*	ضرب	`int z = 2 * 6;` (z برابر 12 می‌شود)
/	تقسیم (با اعشار)	`double a = 15.0 / 3.0;` (a برابر 5.0 می‌شود)
%	باقی‌مانده تقسیم (برای اعداد صحیح)	`int b = 17 % 5;` (b برابر 2 می‌شود)
++	افزایش (افزودن 1)	`int c = 5; ++c;` (c برابر 6 می‌شود)
--	کاهش (کاستن 1)	`int d = 10; --d;` (d برابر 9 می‌شود)

عملگرهای رابطه ای

- عملگرهای رابطه ای برای مقایسه مقادیر استفاده می‌شوند و نتیجه آن‌ها یک مقدار منطقی boolean است که می‌تواند درست true یا نادرست false باشد.

عملگر	نام	مثال
>	بزرگتر	x>y
>=	بزرگتر یا مساوی	x>=y
<	کوچکتر	x<y
<=	کوچکتر یا مساوی	x<=y
==	متساوی	x==y
!=	نامساوی	x!=y

عملگر های رابطه ای در برنامه نویسی C

عملگر های منطقی

- عملگرهای منطقی (Logical Operators) بر روی عبارات منطقی عمل می‌کنند. عبارات منطقی دارای دو ارزش درستی و نادرستی هستند
- در زبان برنامه نویسی c ارزش نادرستی با مقدار ۰ و ارزش درستی با مقادیر غیر صفر مشخص می‌شود.

عملگر	نام	مثال
!	نقض (not)	! y
&&	و (and)	x>y && m<p
||	یا (or)	x>y || m<p

عملگر های منطقی به ترتیب تقدم در برنامه نویسی C

عملگر های ترکیبی

- از ترکیب عملگرهای محاسباتی و علامت = ، مجموعه دیگری از عملگرها ایجاد می شود که عمل محاسباتی و انتساب را انجام می‌دهند. تقدم این عملگرها پایین‌تر از سایر عملگرها می باشد.

عملگر	نام	مثال	معادل
+=	انتساب جمع	x += y	x = x + y
-=	انتساب تفریق	x -= y	x = x - y
*=	انتساب ضرب	x *= y	x = x * y
/=	انتساب تقسیم	x /= y	x = x / y
%=	انتساب باقی مانده تقسیم	x %= y	x = x % y

عملگر های ترکیبی در زبان برنامه نویسی C

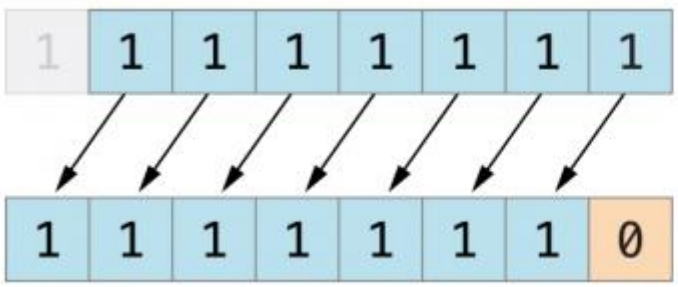
عملگر های بیتی

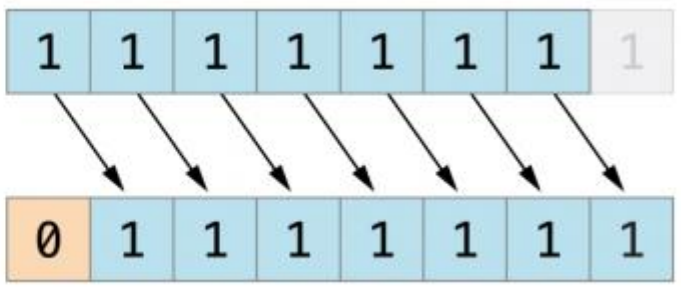
- بر خلاف اکثر زبان ها، زبان برنامه نویسی cروی بیت ها عملیات انجام می دهد. عملگرهای بیتی برای تست کردن، مقدار دادن یا شیفت دادن و سایر اعمال بر روی مقادیری که در یک بایت ((charیا کلمه ((intذخیره شده اند به کار می روند. عملگرهای بیتی را نمی توان با انواع float، double، long doubleوvoid یا سایر انواع پیچیده به کار برد.عملگرهای شیفت، بر روی یک عملوند عمل می‌کنند و بیت های آن را به سمت راست یا چپ شیفت میدهند.

عملگر	نام	
&	AND	و
|	OR	یا
^	XOR	یا انحصاری
~	Not	نقیض
>>	Right shift	شیفت به راست
<<	Left shift	شیفت به چپ

عملگر های بیتی دربرنامه نویسی c

نمایش شیفت و سایر






```
4|3

0 1 0  0 // 4
0 0 1  1 // 3
----------
0 1 1 1 // 7
```

```
15&7

1 1 1 1 // 15
0 1 1 1 // 7
------------
0 1 1 1 // 7
```

```
~8

1 0 0 0  // 8
---------------
0 1 1 1  // 7
```

```
14^9

1 1 1 0  // 14
1 0 0 1   //  9
-----------
0 1 1 1  //  7
```

```
3<<2

0 0 1 1  //  3
---------
1 1 0 0  // 12
```

```
10>>2

1 0  1 0  // 10
----------
0  0  1 0 //  2
```

X	Y	X & Y	X | Y	X ^ Y	~ X
۰	۰	۰	۰	۰	۱
۰	۱	۰	۱	۱	۱
۱	۰	۰	۱	۱	۰
۱	۱	۱	۱	۰	۰

عملکرد عملگرهای بیتی در برنامه نویسی C

عملگرهای متفرقه - عملگر های & , *

- متغیرها نامی برای کلمات حافظه اند و حافظه نیز دارای شماره ردیف می باشد که ما آنها را آدرس می‌نامیم. با استفاده از عملگر & می‌توانیم به آدرس متغیر ها دسترسی داشته باشیم. عملگر * نیز برای دسترسی غیر مستقیم به حافظه مورد استفاده قرار می گیرد. p= & x یعنی آدرس x در p قرار می گیرد.

عملگرهای متفرقه - عملگر ؟

- ابتدا شرط مورد بررسی قرار میگیرد:

```
int x , y ;

x = 5 ;

y = x > 5 ? x * ۲ : x * ۵ ;
```

- متغیر= <شرط> ؟ <عبارت ۱> : <عبارت۲> ;
- اگر ارزش آن درست باشد عبارت ۱ برای انتساب مورد استفاده قرار میگیرد.
- اگر ارزش آن نادرست باشد عبارت ۲ برای انتساب مورد استفاده قرار میگیرد.

عملگرهای متفرقه - عملگر های کاما ,

در حالت های خاصی کاما (,) در نقش یک عملگر کار می کند نه یک جداکننده. یک عبارت را می توان با مربوط کردن دو زیرعبارت توسط کاما شکل داد. هر دو عبارت ارزیابی می شود، ابتدا عبارت سمت چپ محاسبه می شود و نتیجه عبارت سمت راست بعنوان نتیجه کلی عبارت ارزیابی می شود.

متغیر = (< عبارت ۱ > , < عبارت ۲ >) ;

در مثال زیر، a یکی اضافه می کند و مقدار b را در x قرار می دهد سپس آنرا افزایش می دهد.

```
x = (a++ , b++)
```

عملگرهای متفرقه - عملگر sizeof

- عملگر زمان ترجمه است
- طول یک متغیر یا نوع داده را بر حسب بایت تعیین میکند.
- هنگامی که نوع داده مورد استفاده قرار میگیرد باید داخل پرانتز قرار بگیرد.
- sizeofیک تابع نیست و می تواند بدون پرانتز هم استفاده شود.

sizeof متغیر ;

sizeof (نوع داده) ;

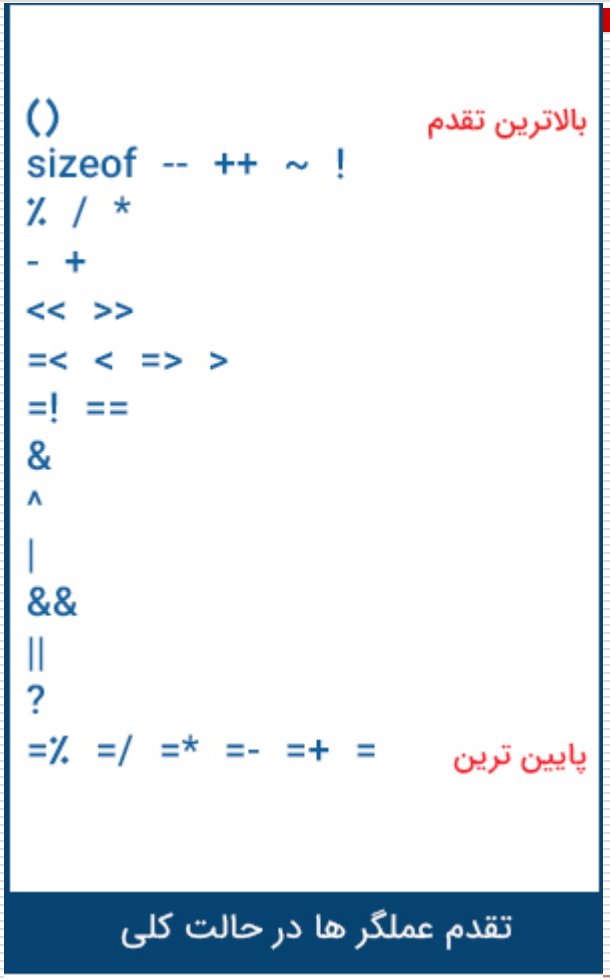
```
int x , y , m ;

x = sizeof y ;

m = sizeof (float);
```

عملگرهای متفرقه - عملگر ()

- پرانتز ها عملگرهایی هستند که تقدم عملگرهای داخل خود را بالا میبرند.



مفهوم پیش پردازنده

- پیش‌پردازنده Preprocessor یک ابزار قدرتمند است که قبل از شروع کامپایل اصلی، کد شما را بررسی و پردازش می‌کند. به شما امکان می‌دهد تا کارهای مختلفی را برای ساده‌سازی، سازماندهی و سفارشی‌سازی کد خود انجام دهید.
- شامل کردن فایل‌های سرآیند: دستورات include به پیش‌پردازنده می‌گویند که کد موجود در فایل‌های دیگر (سرآیندها) را در برنامه شما قرار دهد. این کار برای استفاده از توابع، کلاس‌ها و متغیرهای تعریف شده در فایل‌های دیگر بسیار مفید است.

ساختار برنامه در c++

```
#include <iostream>
using namespace std;
 int main()
{
        اعلان متغیرها
        دستورات اجرایی
    return 0;
}
```

#include

#include یکی از مهم‌ترین دستورات پیش‌پردازنده در زبان برنامه‌نویسی سی++ است.

این دستور به کامپایلر می‌گوید که کد موجود در یک فایل دیگر را به برنامه شما اضافه کند.

این فایل‌ها معمولاً حاوی تعریف توابع، کلاس‌ها، متغیرها و سایر عناصر لازم برای اجرای برنامه هستند.

به آن‌ها فایل‌های سرآیند (Header files) گفته می‌شود.

فضای نام استاندارد std

این فضای نام شامل بسیاری از عناصر استاندارد سی پلاس پلاس مانند cout, cin, string و... است.

برای استفاده از عناصر این فضای نام، معمولاً از دستور using namespace std; در ابتدای برنامه استفاده می‌شود.

مزایا

- **جلوگیری از تداخل نام‌ها: با قرار دادن عناصر در فضاهای نام مختلف، از تداخل نام‌ها جلوگیری می‌شود.**
- **سازماندهی کد: فضاهای نام به شما کمک می‌کنند تا کد خود را به بخش‌های منطقی تقسیم کنید.**
- **استفاده مجدد از کد: می‌توانید فضاهای نام را در پروژه‌های مختلف استفاده کنید.**
- **کاهش احتمال خطا: با استفاده از فضاهای نام، احتمال بروز خطاهای ناشی از تداخل نام‌ها کاهش می‌یابد.**

cout و cin | ابزارهای اصلی ورودی و خروجی

- cout: نمایش اطلاعات روی صفحه نمایش
- از این دستور برای نمایش متن، اعداد و هر نوع داده دیگری روی صفحه نمایش استفاده می‌شود.
- cin: دریافت ورودی از کاربر
- از این دستور برای دریافت اطلاعات از کاربر از طریق صفحه کلید استفاده می‌شود.
- endl برای رفتن به خط بعدی در خروجی استفاده می‌شود.

```cpp
#include <iostream>

using namespace std;

int main() {
    char ch;
    cout << "Enter a character: ";
    cin >> ch;
    cout << "You entered: " << ch << endl;
    return 0;
}
```

کاراکترهای کنترلی

کاراکتر کنترلی	توصیف
\n	ایجاد خط جدید
\t	ایجاد یک تب (فاصله ثابت) 8 محل بعدی
\r	کلید enter را مشخص میکند.
\b	حرکت یک کاراکتر به عقب (backspace) کاراکتر قبل از خودش را حذف میکند.
\\	نمایش بک‌اسلش
\:	نمایش :
\"	نمایش "