Time	Narration
00:01	به spoken tutorial در First C programخوش آمدید
00:05	در این برنامه یاد می گیریم
00:08	چگونه برنامه ساده Cرا بنویسیم
00:11	چگونه آن را کامپایل کنیم
00:13	چگونه آن را اجرا کنیم و تعدادی از اشتباهات رایج و راه حل آن را توضیح می دهیم
00:18 00:21	برای ضبط این برنامه از Ubuntu operating system نسخه 11/10 و gcc Compiler نسخه 4/6/1 در Ubuntu استفاده میکنیم
00:31	برای تمرین این برنامه
00:33	شما باید با Ubuntu operating system و یک Editor آشنا باشید
00:38	بعضی از Editor ها vim و gedit میباشند
00:42	من در اینجا از gedit استفاده میکنم
00:45	برای برنامه آموزشی های مربوطه لطفًا به وب سایت نشان داده شده مراجعه کنید http://spoken-tutorial.org
00:51	با یک مثال نوشتن یک برنامه Cرا نشان می دهم
00:55	با فشار دادن همزمان کلید های Alt, Ctrl و Tروی صفحه کلید یک پنجره ترمینال را باز کنید
01:07 01:12	و حالا یک editor متنی را باز میکنیم سپس gedit فاصله talk نقطه c فاصله & را تایپ کنید.
01:20	از & برای آزاد سازی **prompt** استفاده می کنیم.
01:24	توجه کنید که تمام فایلهای **C** امتداد **c**. را دارند.
01:30	حالا Enter را فشار دهید
01:32	Editor متنی باز شده است
01:36	حالا نوشتن برنامه را آغاز می کنیم
01:39 01:42	// فاصله My first C program (اولین برنامه c من)را تایپ کنید.
01:48	// در اینجا برای comment در خط استفاده می شود
01:52	Comments برای فهمیدن جریان برنامه استفاده می شوند.
01:56	این برای مستناد مفید می باشد
01:58	و به ما در مورد برنامه اطلاعات می دهد.
02:01	Double slash کامنت یک خطی نامیده می شود.
02:07	حالا Enter را فشار دهید
02:09	#Include فاصله کروشه باز و کروشه بسته را تایپ کنید

این یک تمرین خوب می باشدکه در ابتدا کروشه را کامل کنید و سپس شروع به نوشتن در داخل آن کنید.	02:17
حالا در داخل کروشه stdio نقطه h را تایپ کنید.	02:24
Stdio.h یک header file می باشد.	02:30
یک برنامه وقتی که استاندارد input/output functions را استفاده می‌کند باید شامل این header باشد.	02:33
حالا Enter را فشار دهید	02:41
حالا int فاصله main پرانتز باز, پرانتزبسته را تایپ کنید	02:43
, main یک function خاص می باشد.	02:50
این نشان می‌دهد که اجرای برنامه از این خط شروع می شود.	02:52
کروشه باز و کروشه بسته پرانتز نامیده می شود.	02:58
پرانتزهای بعد از main به کاربر نشان می‌دهد که main یک function می باشد	03:04
در اینجا int main function هیچ arguments ندارد.	03:11
و به آن نوع integer (عدد صحیح) را می دهد.	03:15
ما در مورد data type در یک برنامه دیگر یاد می گیریم.	03:18
حالا به اسلاید بعدی برای یاد گرفتن بیشتر main می رویم.	03:23
هر برنامه باید یک main() function داشته باشد.	03:29
نباید بیشتر از یک main function داشته باشیم.	03:33
در غیر این صورت کامپایلر نمی‌تواند شروع برنامه را پیدا کند.	03:36
پرانتزهای خالی نشان می‌دهد که main هیچ arguments ندارد.	03:41
مفهوم arguments را در برنامه بعدی توضیح می دهیم.	03:46
حالا به برنامه خود بازمیگردیم.	03:52
Enter را فشار دهید	03:55
آکولاد راست را تایپ کنید {	03:58
آکولاد راست شروع function main را نشان می دهد	04:00
.سپس آکولاد چپ را تایپ کنید .{	04:04
آکولاد چپ پایان function main را نشان می دهد	04:08
داخل آکولاد Enter را دو بار فشار دهید.	04:13
نشانگر را یک خط بالاتر ببرید.	04:16
Indentation (دندانه گذاری)خواندن کد را آسان‌تر می کند	04:20
و سریع‌تر می‌توان اشتباه را پیدا کرد.	04:23
حالا در اینجا سه فاصله می دهیم.	04:25
حالا printf پرانتز باز و پرانتز بسته () را تایپ کنید.	04:29
Printf یک function استاندارد C برای چاپ خروجی در ترمینال می باشد.	04:34

04:39	که در اینجا داخل کروشه بین double quotes می باشد.
04:43	هر چیزی بین double quotes در دستور printf در ترمینال چاپ می شود.
04:50	Talk To a Teacher\n را تایپ کنید.
04:59	\n خط جدید را نشان می دهد.
05:03	بنابراین بعد از اجرای printf نشانگر به خط جدید می رود.
05:10	هر دستور C باید با ; پایان یابد.
05:15	پس آن را در پایان این خط تایپ کنید
05:19	Semicolon بعنوان پایان دهنده دستور می باشد.
05:24	حالا Enter را فشار دهید و سه فاصله بدهید.
05:27	و return فاصله 0 و یک semicolon را تایپ کنید.
05:34	این دستور عدد 0 را به آن می دهد.
05:38	به این function ما باید عدد صحیح بدهیم چون نوع این function در اینجا int می باشد.
05:45	دستور return پایان دستور قابل اجرا را نشان می دهد.
05:51	مقادیر بازگشتی را در برنامه دیگر بیشتر یاد می کیریم.
05:55	حالا برای ذخیره فایل save را کلیک کنید.
06:00	عادت خوبی است که دائماً فایلها را ذخیره کنید.
06:03	این شما را از قطع ناگهانی برق محافظت می کند
06:05	و در صورت سقوط application نیز مفید می باشد.
06:10	حالا برنامه را کامپایل می کنیم. به ترمینال باز می گردیم
06:15	و gcc فاصله talk.c فاصله hyphen -o فاصله my output را تایپ کنید.
06:24	کامپایلر gcc می باشد
06:27	فایل ما talk.c می باشد.
06:30	o- myoutput میگوید که اجرا باید به فایل myoutput برود.
06:37	Enter را فشار دهید
06:39	می‌بینیم که برنامه کامپایل شده است
06:42	با تایپ کردن ls space (hyphen) -lrt می‌بینیم که myoutput آخرین فایلی است که باید ایجاد شود.
06:54	برای اجرای برنامه myoutput/.را تایپ و Enter را فشار دهید
07:01	در اینجا خروجی Talk To a Teacher نمایش داده می شود
07:06	همان‌طور که گفتم return آخرین دستوری می‌باشد که باید اجرا شود
07:10	بعد از return چیز دیگری اجرا نمی شود. حالا آن را انجام می دهیم.
07:15	به برنامه‌مان بر می گردیم
07:17	بعد از دستور return یک دستور printf دیگر را نیز شامل می کنیم

اینجا فاصله بدهید و printf کروشه باز , کروشه بسته را تایپ کنید	07:22
داخل کروشه بین double quotes این welcome backslash n را تایپ و در آخر semicolon را تایپ کنید.	07:27
حالا save را کلیک کنید	07:35
و come back را در ترمینال کامپایل و اجرا کنید.	07:37
کامنت هایی را که قبلاً وارد کرده‌ایم را با استفاده از کلید up arrow می‌توانیم دوباره ببینیم	07:41
و این کاری است که من حالا کردم.	07:46
می‌بینیم که بیانیه دوم welcome اجرا نمی شود	07:51
حالا به برنامه‌مان بر می گردیم	07:58
و بیانیه welcome را بالای بیانیه return می نویسیم	08:00
Save را کلیک کنید.	08:06
و کامپایل و اجرا می کنیم.	08:09
می‌بینیم که دومین printf بیانیه welcome نیز اجرا می شود.	08:15
حالا اشتباهات رایج را می‌بینیم , به برنامه خود بر می گردیم.	08:23
فرض کنید که در اینجا من نقطه در stdio.h را نگذاشته ام. Save را کلیک کنید.	08:29
حالا کامپایل و اجرا می کنیم.	08:36
می‌بینیم که اشتباه مهلک در خط شماره 2 در فایل talk.c وجود دارد.	08:41
کامپایلر نمی‌تواند header file با نام stdioh را پیدا کند بنابراین اشتباه "no such file or directory" داده می شود.	08:48
بنابر این مجموعه خاتمه داده می شود.	08:59
حالا اشتباه را درست می کنیم.نقطه را "." دوباره وارد کنید و save را کلیک کنید	09:03
حالا کامپایل و اجرا می‌کنیم و اجرا می شود.	09:11
حالا یک اشتباه رایج دیگر را نشان می دهم.	09:19
دوباره به برنامه بر می گردیم.	09:22
حالا فرض می‌کنیم که semicolon در آخر خط را نکذاشته ام.	09:25
Save را کلیک کنید و آن را کامپایل و اجرا می کنیم	09:31
می‌بینیم که اشتباه در خط ششم در فایل talk.c وجود دارد که semicolon قبل از printf را انتظار دارد.	09:41
به برنامه بر می گردیم.	09:51
همان‌طور که گفتم semicolon پایان دهنده بیانیه می باشد.	09:54
بنابراین در پایان خط 5 و شروع خط 6 آن را جستجو می کند.	09:58
این خط 6 می باشد.	10:06
این آخرین جایی است که می‌توانید semicolon را بگذارید.	10:09
همچنین کامپایلر پیام اشتباه در خط 6 را می دهد.	10:12
ببینیم اگر semicolon را در اینجا بگذاریم چه می شود.	10:18

10:23	Save را کلیک کنید
10:26	حالا کامپایل و اجرا می کنیم.
10:30	بله, اجرا می شود.
10:32	به برنامه برمی‌گردیم در آخر خط 6, semicolon را تایپ می کنیم.
10:40	این یک تمرین خوب می‌باشد که semicolon را در آخر خط تایپ کنیم.
10:46	Save را کلیک کنید
10:49	حالا کامپایل و اجرا می کنیم.و اجرا می شود.
10:54	به اسلاید بر می گردیم.
10:57	و حالا ارائه:
10:59	یک برنامه برای پرینت "Welcome to the World of C" بنویسید
11:02	ببینید اگر n\ را در دستور printf ننویسیم چه می شود.
11:08	این ما را به پایان برنامه آموزشی می آورد
11:12	ویدیو را در لینک زیر مشاهده کنید.
11:15	این خلاصه پروژه Spoken Tutorial میباشد
11:18	اگر پهنای باندخوبی ندارید، ابتدا دانلود وسپس مشاهده کنید
11:22	تیم پروژه ‘’’Spoken Tutorial’’’
11:24	کارگاه آموزشی استفاده از ‘’’Spoken Tutorial’’’ ارائه میدهد
11:28	و به کسانی که آزمون آنلاین را قبول شوند گواهینامه میدهد
11:31	برای جزئیات بیشتر لطفاً به’’’contact@spoken-tutorial.org’’’ ایمیل بفرستید
11:38	Spoken tutorial بخشی از پروژه Talk To a Teacher می باشد.
11:42	که تحت پشتیبانی National Mission on Education از طریق ICT ، توسط MHRDدولت هند می باشد
11:47	اطلاعات بیشتر درلینک زیر موجود می باشد ’’’spoken hyphen tutorial dot org slash NMEICT hyphen Intro’’’
11:51	با تشکر از شما. شبنم اقبال خان