Time	Narration
00:01	'Storage class specifiers' پراسپوکن ٹیوٹوریل میں خوش آمدید.
00:07	اس ٹیوٹوریل میں ہم کچھ مثالوں کی مدد سے مندرجہ ذیل کے بارے میں سیکھیں گے 'Storage class specifiers' 'auto keyword' 'static keyword' 'extern keyword' 'register keyword'
00:22	اس ٹیوٹوریل کے لئے میں 'ابنٹو آپریٹنگ سسٹم ورشن '11.10 اور 'ابنٹو پر gcc کمپائلر ورشن '4.6.1 استعمال کر رہا ہوں
00:34	اس ٹیوٹوریل کو سمجھنے کے لئے آپ کو C ٹیوٹوریلس سے واقف ہونا چاہئے.
00:41	اگر نہیں تو متعلقہ ٹیوٹوریلس کے لئے دکھائی گئی ہماری ویب سائٹ پر جائیں.
00:47	میں 'storage class specifiers' کے تعارف کے ساتھ شروع کروں گا.
00:52	'Specifiers' * کمپائلر کو بتاتے ہے کہ 'ویریبل ' کو کہاں جمع کیا کرنا ہے.
00:57	variable کو کیسے سیو کرتے ہیں.
00:59	variable کی ابتدائی ویلیو کیا ہوتی ہے.
01:03	variable کا لائف ٹائم
01:06	سِنٹیکس ہے: 'storage_specifier data_type variable _name'
01:13	اسٹوریج کلاس سپیسپفائر کے مندرجہ ذیل اقسام ہیں: 'auto' 'static' 'extern' 'register'
01:21	اب auto کی وارڈ کے ساتھ شروع کرتے ہیں.

01:24	auto کی وارڈ automatic variable ڈکلیئر کرتا ہے.
01:28	یہ local دائرہ کار رکھتا ہے.
01:30	keywords خود بخود انشلائز نہیں ہوتے ہیں.
01:34	ڈکلیئر کرتے وقت، keywords کو وضاحت سے انشلائز کرنا چاہیئے.
01:39	keywords کی اسٹوریج اسپیس CPU memory ہوتی ہے.
01:43	اب ایک مثال دیکھتے ہیں . میرے پاس ایک کوڈ فائل ہے؛ اب اسے دیکھتے ہیں.
01:49	نوٹ کریں ہماری فائل کا نام 'auto.c' ہے.
01:54	ہم نے 'increment' کے طور پر ایک فنکشن کو ڈکلیئر کر لیا ہے.
01:58	یہ main() ہے.
02:00	main() میں، increment() فنکشن چار بار کال ہوتا ہے.
02:06	پھر ہمارے پاس 'return 0' اسٹیٹمینٹ ہے.
02:10	اب فنکشن کی وضاحت دیکھتے ہیں.
02:14	یہاں ہم نے ویریبل 'i' کو 'auto int' کے طور پر ڈکلیئر کیا ہے . یہ local دائرہ کار رکھتا ہے.
02:21	پھر ہم 'printf' استعمال کر کے 'i' کی ویلیو کی عکاسی کرتے ہیں.
02:26	یہاں 'i' کی ویلیو بڑھائی جاتی ہے.
02:30	اب اپنے کی بورڈ پر ایک ساتھ Ctrl + Alt + T' کیز دبا کر terminal کھولیں.
02:38	ٹائپ کریں: 'gcc space auto.c space hyphen o space auto'. ، اینٹر دبائیں.
02:48	ٹائپ کریں: dot slash auto
02:51	آؤٹ پٹ زیرو ہے.
02:54	اب اپنے پروگرام پر واپس آتے ہیں.
02:57	اب main() کے اوپر 'auto variable i' کو انشلائز کرتے ہیں.
03:02	میں اس ڈکلیئریشن اور 'انشلائزیشن کو یہاں سے کٹ کروں گا اور یہاں پر پیسٹ کروں گا. اب 'Save' پر کلک کروں گا.
03:14	اب terminal پر ایکزِیکیوٹ کرتے ہیں . اپ ایرو کی دو بار دبائیں. اینٹر دبائیں.

03:22	ہمیں ایک ایرر ملتا ہے: 'file-scope declaration of i specifies auto'
03:29	ایسا اس لئے ہے کیونکہ ایک auto ویریبل، function کے لئے local ہے.
03:34	ہم اسے گلوبلی، initialize نہیں کر سکتے.
03:37	اب ایرر کو درست کرتے ہیں . اپنے پروگرام پر واپس آتے ہیں.
03:42	اسے ڈِلیٹ یعنی حذف کریں؛ اور اسے یہاں پیسٹ کریں.
03:47	'Save' پر کلک کریں اور terminal پر ایکزِکیوٹ کریں.
03:52	اپ ایرو کی دبائیں . پچھلی کمانڈ کو دوبارہ کال کریں.
03:57	اینٹر دبائیں . ٹائپ کریں dot slash auto ، اینٹر دبائیں.
04:03	یہ کام کر رہا ہے ! آؤٹ پٹ زیرو ہے.
04:07	ایسا اس لئے ہے کیونکہ ہم نے 'i' کی ویلیو 0 سے انشلائز کی ہے.
04:13	اب 'static variable' دیکھتے ہیں.
04:16	اگرچہ ہم گزشتہ ٹیوٹوریل میں 'static variable' کے بارے میں پڑھ چکے ہیں . میں یہاں اسے مختصر میں سمجھاوں گا.
04:24	'static' ویریبلس zero سے انشلائز ہوتے ہیں.
04:28	block سے پروگرام کنٹرول ایکزِکیوٹ ہونے کے بعد بھی یہ تباہ نہیں ہوتے ہیں.
04:35	variable کی ویلیو مختلف function calls کے درمیان قائم رہتی ہے.
04:41	اسٹوریج اسپیس، CPU memory ہے.
04:45	اب ایک مثال دیکھتے ہیں . میں وہی کوڈ فائل ایڈٹ کروں گا.
04:51	اپنے پروگرام پر واپس آتے ہیں.
04:54	ایک ساتھ 'Ctrl + Shft + S' کیز دبائیں.
05:01	اب میں فائل کا نام 'static' کروں گا، اور 'Save' پر کلک کروں گا.
05:10	اب، میں 'variable i' کے انشلائزیشن کو بدل کر 'static int i equals to zero' کروں گا. اور 'Save' پر کلک کروں گا.
05:23	اب دیکھتے ہیں کہ کیا ہوتا ہے' . ٹرمنل 'پر فائلوں کو ایکزِکیوٹ کرتے ہیں.
05:30	ٹائپ کریں: 'gcc space static.c space hyphen o space stat' ، اینٹر دبائیں.

05:41	ٹائپ کریں dot slash stat۔ انٹر دبائیں.
05:46	آؤٹ پٹ مندرجہ ذیل طور پر ظاہر ہوتا ہے : "0, 1, 2, 3"
05:51	ایسا اس لئے ہے کیونکہ 'static variables' ، 'global variables' ہیں.
05:56	'static variable' کا دائرہ کار function کے لئے local ہے، جس میں وہ ڈفائن کیا گیا ہے.
06:03	یہ function calls کے درمیان اپنی ویلیو کو نہیں کھوتے
06:08	اب 'extern keyword' کے بارے میں سیکھتے ہیں.
06:12	'extern variable' کا دائرہ کار پورے مین پروگرام میں ہے.
06:17	'extern variable' کی وضاحت 'C' پروگرام میں کہیں بھی ہو سکتی ہے.
06:23	ڈیفالٹ طور پر، 'extern variables' زیرو سے انشلائز ہوتے ہیں.
06:28	یہ پروگرام میں سارے functions کیلئے قابل رسائی ہوتے ہیں.
06:33	یہ 'CPU memory' میں سٹور یعنی جمع ہوتے ہیں.
06:36	اب ایک مثال دیکھتے ہیں.
06:38	میرے پاس ایک کوڈ فائل ہے؛ اب اسے دیکھتے ہیں.
06:42	نوٹ کریں ہماری فائل کا نام 'extern.c' ہے.
06:47	میں نے x نامی ایک ویریبل کو integer variable ، '10' سے انشلائز کیا ہے.
06:54	یہ main() ہے . main() میں، میں نے ایک 'extern integer variable y' ڈکلئیر کیا ہے.
07:03	'printf' اسٹیٹمینٹ استعمال کرتے ہوئے ہم 'x' اور 'y' کی ویلیوز ظاہر کریں گے. یہ 'return' اسٹیٹمینٹ ہے.
07:12	main() کے بند ہونے کے بعد ہم 'y' کو '50' سے انشلائز کریں گے.
07:18	اب terminal کھولتے ہیں اور دیکھتے ہیں کہ آؤٹ پُٹ کیا ہوگا.
07:24	ٹائپ کریں: 'gcc space extern.c space hyphen o space ext'. ، انٹر دبائیں.
07:35	ٹائپ کریں : dot slash ext ، انٹر دبائیں.
07:40	آؤٹ پٹ مندرجہ ذیل طور پر نظر آئے گا: 'The value of x is 10' 'The value of y is 50'
07:48	جیسا کہ ہم نے پڑھا ہے 'extern keyword' کی ویلیو مین پروگرام میں شروع سے آخر تک ہوتی ہے.
07:55	پروگرام میں کہیں بھی اس کو ڈفائن یعنی وضاحت کر سکتے ہیں.

07:59	دونوں اسٹیٹمینٹس صحیح ہوتے ہیں.
08:02	اب 'register keyword' پر جاتے ہیں.
08:06	'Register' ویریبلس تک رسائی normal ویریبلس کے مقابلے میں جلدی ہوتی ہے۔.
08:13	یہ 'main memory' کے بجائے 'register memory' میں جمع ہوتے ہیں.
08:19	ویریبلس کی محدود تعداد استعمال کی جا سکتی ہے کیوں کہ register size بہت کم ہے.
08:25	16 bits، 32 bits یا 64 bits
08:30	اب ایک مثال دیکھتے ہیں . میرے پاس ایک کوڈ فائل ہے .اب اسے دیکھتے ہیں.
08:37	نوٹ کریں، فائل کا نام 'register.c' ہے.
08:42	یہاں ہم نے 'register integer variable' ڈکلئیر کیا ہے.
08:47	یہ variable براہ راست register memory میں اسٹور یعنی جمع ہوگا.
08:53	یہ 'for loop' ہے جو '1' سے '5' تک 'i' کی ویلیو ظاہر کرتا ہے.
08:59	یہ 'i' کی ویلیو دکھائے گا.
09:03	اب پروگرام کو ایکزِ کیوٹ کرتے ہیں اور دیکھتے ہیں.
09:07	terminal پر، ٹائپ کریں، 'gcc space register.c space hyphen o space register'
09:17	اینٹر دبائیں . ٹائپ کریں dot slash register ، اینٹر دبائیں.
09:25	آپ مندرجہ ذیل طور پر ظاہر آؤٹ پٹ دیکھ سکتے ہیں : 'Values stored in register memory 1 2 3 4 5'
09:34	ہم اس ٹیوٹوریل کے آخر میں آ گئے ہیں .اس کا خلاصہ بیان کرتے ہیں.

09:39	اس ٹیوٹوریل میں ہم نے سیکھا: 'Storage class specifiers' 'auto keyword' 'static keyword' 'extern keyword' 'register keyword'
09:52	ایک مشق کے طور پر، پہلے پانچ نمبرس کے جوڑ کو پرنٹ کرنے کے لئے ایک پروگرام لکھیں
09:59	پروگرام میں 'auto' اور 'static' دونوں کی وارڈس کو ڈکلیئر کریں.
10:04	مندرجہ لنک پر دستیاب ویڈو دیکھیں.
10:07	یہ اسپوکن ٹیوٹوریل پروجیکٹ کا خلاصہ بیان کرتا ہے.
10:11	اچھی بینڈوڈتھ نہ ملنے پر آپ اسے ڈاؤن لوڈ کر کے دیکھ سکتے ہیں.
10:16	اسپوکن ٹیوٹوریل پروجیکٹ ٹیم اسپوکن ٹیوٹوریلس کا استعمال کرتے ہوئے ورکشاپ چلاتی ہے.
10:22	آن لائن ٹیسٹ پاس کرنے والوں کو سرٹیفکیٹ دیتے ہیں . مزید معلومات کے لئے، contact@spoken-tutorial.org پر لکھیں.
10:33	اسپوکن ٹیوٹوریل پروجیکٹ ٹاک ٹو اے ٹیچر پراجیکٹ کا حصہ ہے.
10:38	یہ بھارتی حکومت کے ایم ایچ آر ڈی کے آئی سی ٹی کے ذریعے قومی خواندگی مشن کی طرف سے حمایت شدہ ہے.
10:45	اس مشن پر مزید معلومات http://spoken-tutorial.org\NMEICT-Intro پر دستیاب ہیں.
10:52	اس ٹیوٹوریل کا ترجمہ اور صدا بندی میں نے یعنی وجاہت احمد نے کی ہے . شامل ہونے کے لئے شکریہ.