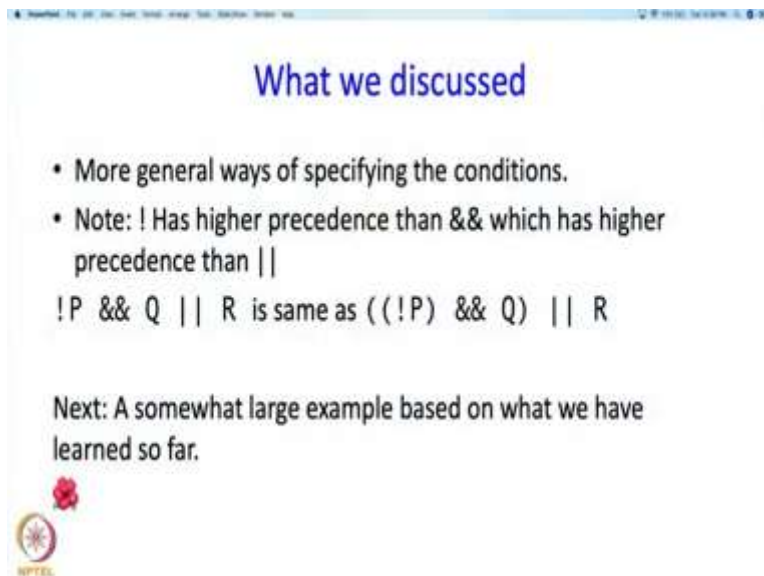


An Introduction to Programming through C++
Professor Abhiram G. Ranade
Department of Computer Science and Engineering
Indian Institute of Technology Bombay
Lecture No. 6 Part – 4
Conditional Execution
A somewhat large program example

(Refer Slide Time: 00:22)




What we discussed

- More general ways of specifying the conditions.
- Note: ! Has higher precedence than && which has higher precedence than ||

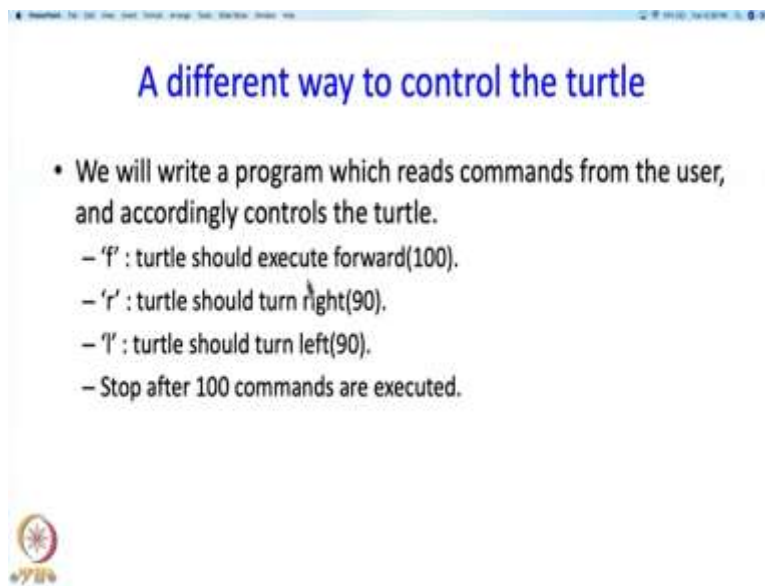
`!P && Q || R` is same as `((!P) && Q) || R`

Next: A somewhat large example based on what we have learned so far.



Welcome back. In the previous segments we discussed general ways of specifying conditions. In this segment we are going to take a large, a large-ish program example. It is not going to be its actually going to be small but it will actually do something in that sense it is a large example, it is a it is a full program which does something.

(Refer Slide Time: 00:39)



A different way to control the turtle

- We will write a program which reads commands from the user, and accordingly controls the turtle.
 - 'f' : turtle should execute `forward(100)`.
 - 'r' : turtle should turn `right(90)`.
 - 'l' : turtle should turn `left(90)`.
 - Stop after 100 commands are executed.



Ok, so what we want to right is the different way of controlling turtle. What do I mean by this? Well here is here is what I mean. So write a program which reads commands from the user, and accordingly controls the turtle. So, the way the turtle moves is not specified in the program, but the user tells how the turtle moves by typing in programs. So if the user types 'f' then the turtle should execute `forward(100)`, if the user types 'r' than the turtle should turn `right(90)`. If the user types 'l' the turtle should turn left by 90 degrees. Ok this are just rules I made up. Ok you could have made a different set of rules and in you will be encouraged to make a more interesting or nicer rules later on in exercises. And this whole thing should stop when 100 commands are executed. Why hundred? Again just a number I made up.

(Refer Slide Time: 01:47)

The program

```
main_program{
char command;
turtleSim();

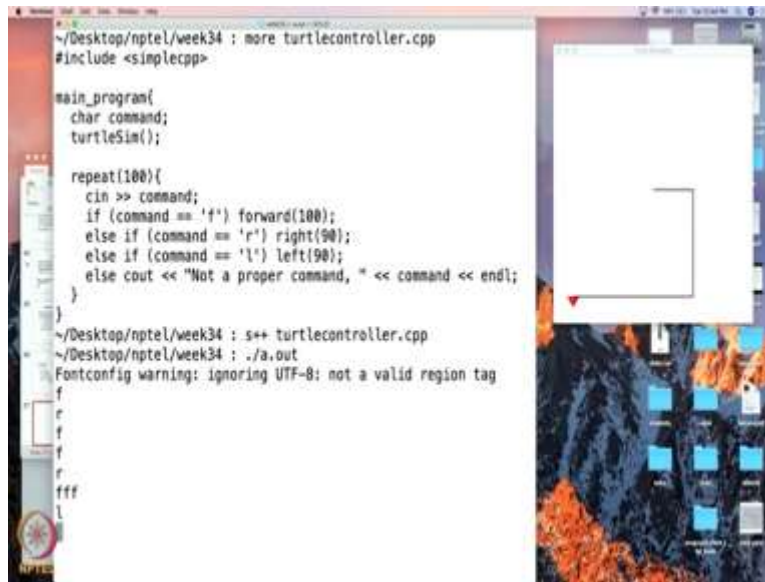
repeat(100){
cin >> command;
if (command == 'f') forward(100);
else if (command == 'r') right(90);
else if (command == 'l') left(90);
else cout << "Not a proper command, "
<< command << endl;
}
}
```



So again this program by now should appear reasonably, reasonably obvious. So, we are going to have a character which will call command and then we will start the turtle, turtle simulator and then since we want to execute 100 commands then we will have a loop with 100 which will repeat 100 times. Then we will read in each iteration we will read in into command. So, depending upon what the user types command will have that value.

If, the user type f and f went to into command, then we will execute forward(100). If, the user type r then we will execute right(90), if, the user type l, we execute left(90) otherwise and when will this be executed? Well if the user typed some other letter may be the user typed g in that case we will help out the user by saying look you typed something which is not a proper command, ok? And we will tell user what command he or she typed. And this will repeat 100 times and that will be the end of the program.

(Refer Slide Time: 03:16)



```
~/Desktop/nptel/week34 : more turtlecontroller.cpp
#include <simplecpp>

main_program{
char command;
turtleSim();

repeat(100){
cin >> command;
if (command == 'f') forward(100);
else if (command == 'r') right(90);
else if (command == 'l') left(90);
else cout << "Not a proper command, " << command << endl;
}
}

~/Desktop/nptel/week34 : s++ turtlecontroller.cpp
~/Desktop/nptel/week34 : ./a.out
Fontconfig warning: ignoring UTF-8: not a valid region tag
f
r
f
f
r
f
f
f
l
```

Alright so, let us do the demo of this. So ok what is this program? This is the program that I have just showed you ok nothing else. So let us compile it ok. Ok so let me just type out the program. So this is the program that I showed you. And now let me compile it. And now let us run it. Let me adjust it. So now let us say we type f. So the turtle has moved forward hundred. So let us say I typed r, the turtle has turned right, maybe I type an f again it has type it has gone forward maybe I type another f, it has gone forward further. So, maybe I will hit r again so it has turned right and I will type f and instead of hitting, hitting enter immediately maybe I will type f two times or three times let us see what happens. So it has actually taken three commands, ok? So, so it has done f three times ok? And may I can turn left here, so it has turned left and it can go on in this manner. So I am not going to go at 100 times but you, you see the program is doing exactly what we wanted. So, let me break out of it. Ok, so we just did this demo and let me now give you an exercise.

(Refer Slide Time: 05:45)

Exercise

- Write a program that reads a number and prints 1 if the number is a multiple of 5 but not of 3, and otherwise prints 0. Write this in as many different ways as possible.
 - Using only simple conditions, e.g. expression 1 == expression 2, but with if statements nested inside each other.
 - Using a single if-then-else statement with a compound condition.



So the exercise is: write a program that reads a number and prints 1 if the number is a multiple of 5 but not 3 and otherwise prints 0. And you are encouraged to write this in as many as ways as possible ok? So for example, you could write this using only simple conditions. So for example, you could write something like, you could only have conditions of the form expression 1 equal to equal to expression 2 or whatever, whatever or not equal to greater than or equal to or so on ok?

But here, you may have to put if statements nested inside one another ok so inside say a consequent or alternate u may have an if statement, ok? Or, you can use a single if then else statement with a compound condition ok? So this is really and exercise for learning the if statement and different ways of expressing the same program using different possibilities that that language gives you.

(Refer Slide Time: 07:03)



Alright, so, what did we discussed in this segment? We discussed a detailed program a detailed program example. Next, we are going to discuss the switch statement and logical data but let us take a quick break.