

Control Structures

Aim: How can we use control structures to write more powerful programs?

Goals

- Observe how control structures can make our programs more interesting
- Learn the “while,” “if/else,” and “do/while” control structures
- Become familiar with how mathematical comparisons are expressed in NQC
- Interpret programs that contain these various aspects

So Far in NQC...

- Basic Motor Commands
- Advanced Motor Commands
- Programming with Constants
- Programming with Variables
- Repeat function

*Chapters 1-3 in the NQC tutorial

Control Structures

- Control structures are statements that control the way that other statements are executed
- Examples of Control Structures
 - Repeat
 - While
 - If...Else
 - Do...While

Repeat

- Simply causes a sequence of commands to be executed a predetermined number of times
- Repeated commands are surrounded by brackets
- Similar to loops we used in Robolab

Square Example: Repeat

```
#define FWD_TIME 100
#define TURN_TIME 75

task main()
{
    repeat(4)           // Task executes 4 times
    {
        OnFwd(OUT_A + OUT_C);    Wait(FWD_TIME);
        Rev(OUT_C);              Wait(TURN_TIME);
    }
    Off(OUT_A + OUT_C);
}
```

While Statement

- “While” causes a set of commands to execute as long as a certain condition is true
- It gets used as follows:

```
while (condition)
{ blah blah blah; }
```

Special Conditions –True and False

- True is a “special word” that is always true
- False is a “special word” that is never true
- These can be used as conditions in the while command
- Ex:

```
while (true)
{
    OnFwd(OUT_A + OUT_C);
    Wait(60);
}
```

Rewriting the Square Ex. with “while”

- We input the word “true” as the while condition
- What is happening in the program?
- For how long will the motors be running?

Square Example: while (true)

```
#define FWD_TIME 100
#define TURN_TIME 75

task main()
{
    while(true)
    {
        OnFwd(OUT_A + OUT_C);    Wait(FWD_TIME);
        Rev(OUT_C);              Wait(TURN_TIME);
    }
}
```

Rewriting the Square Ex. with “while”

- We now input the word “false” as the While condition
- What happens when we run the program?

Square Example: while (false)

```
#define FWD_TIME 100
#define TURN_TIME 75

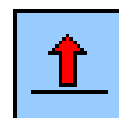
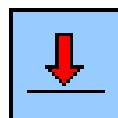
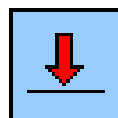
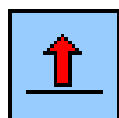
task main()
{
    while(false)
    {
        OnFwd(OUT_A + OUT_C);    Wait(FWD_TIME);
        Rev(OUT_C);              Wait(TURN_TIME);
    }
}
```

Robolab Equivalency

- What Robolab icons do while(true) and while(false) remind us of?

Jump and Land

Land and Jump



Other Conditions in while()

- True and False are extreme cases
 - True is always true, false is always false
- Their truth values never change
- Conditions can also be numerical comparisons
- Same idea except that the truth of the condition can change

Rewriting the Square Ex. with “while”

- We now define a variable called *loops*, and increment it each time the task is executed
- Our while condition is that `loops < 4`
- What happens when we run the program?
- How many times does the task loop before the motors turn off?

Square Example: while()

```
int power = 0;

task main()
{
    while(power <= 7)
    {
        SetPower(OUT_A + OUT_C, power);
        OnFwd(OUT_A+OUT_C);
        Wait(100);
        power += 1;
    }
    Off(OUT_A + OUT_C);
}
```

If/Else Statement

- Used when you want robot to respond different ways depending on a certain condition
- You set up some sort of condition:
 - If it's true, one set of commands will run
 - If it's not true, another set of commands will run

Using If/Else Statement

- Basic setup:

```
if (condition is true)
{blah blah blah blah;}
else
{blah blah blah blah;}

```

Example: If/Else

- What is happening in this program?
- What do you notice about the “if” condition?
- How many times does the “if” condition get checked?

```
#define FWD_TIME 100
#define TURN_TIME 75
int UGLY;

task main
{

    OnFwd(OUT_A + OUT_C);      Wait(FWD_TIME);
    UGLY = Random(1);
    if(UGLY == 0)
    {Rev(OUT_C);}              // Makes a right turn

    else
    {Rev(OUT_A);}
    Wait(TURN_TIME);          // Makes a left turn
    Off(OUT_A + OUT_C);
}

```

Checking the Condition More

- How can we check the “if” condition more than once?

Putting If/Else in a While Loop

Comparison Statements

- In our “if” condition, we used a ‘==’ (double equals sign)
–If (Random(1) == 0)
- Double equals is used to distinguish this statement from a variable definition (ex: TURN_TIME = 60;)
- We use a number of other operators to make comparisons

```

task main()
{
    while(true)
    {
        OnFwd(OUT_A + OUT_C);
        Wait (100);
        int UGLY = Random(1);

        if(UGLY == 0)
        {Rev(OUT_C);}

        else
        {Rev(OUT_A);}

        Wait(60);
    }
}

```

Comparison Symbols

Symbol	Meaning
==	Equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
!=	Not equal to

Using Comparisons in Conditions

```

if(Random(3) < 2)
{
    OnRev(OUT_C);
}
else
{
    OnRev(OUT_A);
}
Wait(TURN_TIME);

```

```

if(Random(10) >= 5)
{
    OnRev(OUT_C);
}
else
{
    OnRev(OUT_A);
}
Wait(TURN_TIME);

```

Combining Conditions

- You can combine more than one condition by using the following symbols:

Symbol	Meaning
&&	And
	Or

- ‘|’ is made by holding down Shift and pressing \

Examples of Conditions

Statement	Meaning
AAA != 3	True when variable ‘AAA’ does not equal 3
(AAA >= 5) && (AAA <10)	True when AAA is greater than or equal to 5 and less than 10
(AAA == 10) (BBB == 10)	True when AAA or BBB equals 10

Example: Combining Conditions

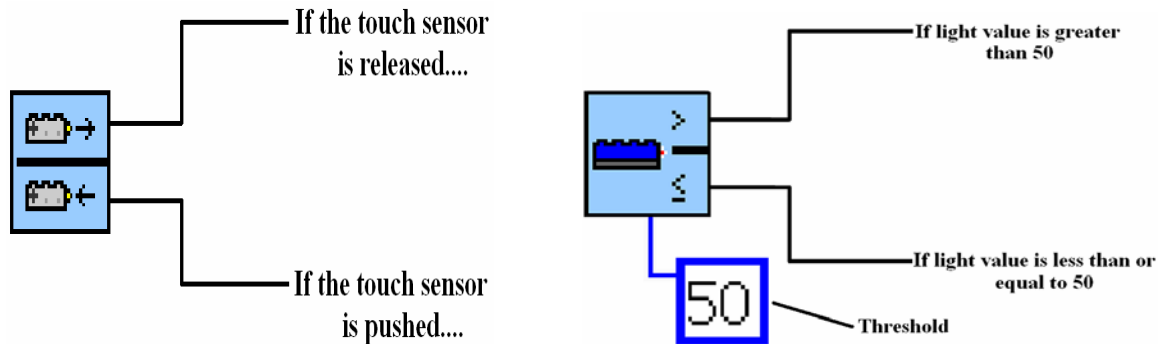
```
task main()
{
    while(true)
    {
        int STR = Random(60);

        if(STR <= 20 || STR >= 40)
        {OnFwd(OUT_A + OUT_C);
        Wait(20);}

        else
        {OnRev(OUT_A + OUT_C);
        Wait(20);}
    }
}
```

Robolab Equivalency

- What Robolab icons does the if/else structure remind us of?
- Light, Touch, Temp Sensor Forks



Do/While Statement

- Very similar to While statement
 - Condition is checked at the end of the task
- Has the following setup:

```
do
{
  blah blah blah blah blah;
}
while (condition);    // semicolon
```
- The statements execute as long as the condition is true

Example: Condition is False

```
int snakes_on_a_plane = 5;

task main()
{
  SetPower(OUT_A + OUT_C, 6);
  do
  {
    OnFwd(OUT_A + OUT_C);
    Wait(200);
  }
  while (snakes_on_a_plane > 7); // Loops while this is true

  Off(OUT_A + OUT_C);
}
```

Example: Do/While Statement

```
int FWD_TIME, REV_TIME, TOTAL_TIME;

task main()
{
    TOTAL_TIME = 0;
    do
    {
        FWD_TIME = Random(100);
        REV_TIME = Random(100);

        OnFwd(OUT_A + OUT_C);           Wait(FWD_TIME);

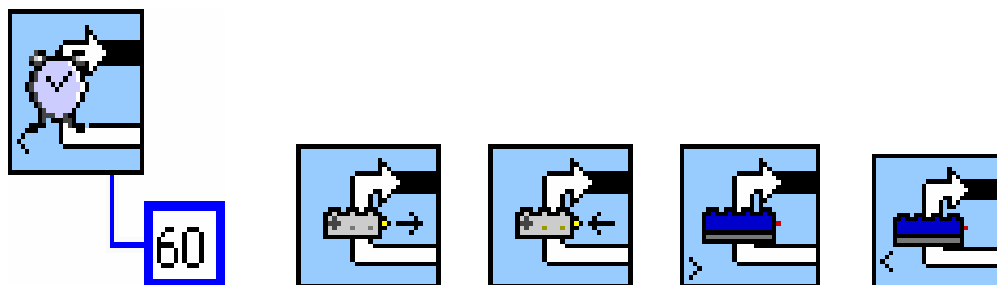
        OnRev(OUT_A + OUT_C);          Wait(REV_TIME);

        TOTAL_TIME += FWD_TIME;
        TOTAL_TIME += REV_TIME;
    }
    while(TOTAL_TIME < 1000);

    Off(OUT_A + OUT_C);
}
```

Robolab Equivalency

- What Robolab icons does the do/while structure remind us of?
- Timer, Light, Touch Sensor Loops



Summary

- Control structures can be used to produce more powerful programs
- Repeat is used to loop a certain task
- While (true) can create an infinite loop
- If/Else will execute code based on whether or not a condition is true
- Do/While will execute code as long as a particular condition is true