Programming Principles

Unit 4

Selection

Selection

Making choices within a program

Selection

- **Selection** is the third fundamental concept in program design
- along with sequence and iteration
- Your program does not always have to follow the same path
- Decisions can be made to go one way or another, depending on a condition

This week we shall use:

- the **if(…)** statement
- the **if(…) else ..** statement

if(…)

Choose to do something or leave it

Syntax (grammar rules)

The **if(…)** statement allows a single choice to be made

```
extern void object:: ifProgram()
{
        //   A :  the beginning of program goes here

            if ( condition )
            {
                // B :
                // code here is done once if condition is true.
                // if condition is not true, the program skips
                // to C below
            }

        //   C :  this part of the program is done after the if() statement
        //   so there are 2 paths:  A-B-C (condition true) or A-C (false)
}
```

a **condition** is either **true** or **false**

An Example

Butter up the lecturers

```
extern void object:: ChatUpLecturer()
{
        string response;
        response = dialog("Hello .. are you a student or a lecturer?");
        if ( response == "lecturer" )
        {
            message ("You look too young to be a lecturer");
            message ("I was sure you were a student!");
        }
        message ("Well I'd better get on …");
        if ( response == "lecturer" )
        {
            message (" … with my studies");
        }
}
```

What are the possible results?

2 possible conversations

The condition true path

Hello .. are you a student or a lecturer?

lecturer

You look too young to be a lecturer I was sure you were a student!

Well, I'd better get on …

… with my studies!

OR

The condition false path

Hello .. are you a student or a lecturer?

student

Well, I'd better get on …

Variation

Slag off the lecturers

```
extern void object:: SlagOfflecturer()
{
        string response;
        response = dialog("Hello .. are you a student or a lecturer?");
        if ( response != "lecturer"  )
        {
            message ("Thank God .. lecturers are so boring and stupid");
            message (" … and so ugly!");
        }
        message ("Well I'd better get on …");
        if ( response != "lecturer"  )
        {
            message (" … down to the bar!");
        }
}
```

What are the results this time?

2 possible conversations

The condition true path

Hello .. are you a student or a lecturer?

student *(or anything else)*

Thank God … lecturers are so boring and stupid … and so ugly!

Well, I'd better get on …

… down to the bar!

OR

The condition false path

Hello .. are you a student or a lecturer?

lecturer

Well, I'd better get on …

What would happen if you responded with **Lecturer** instead of lecturer?

||
(the logical OR operator)

Combining conditions

An Example using || (OR)

Butter up the lecturers .. better version

```
extern void object:: ChatUplecturerORLecturer()
{
        string response;
        response = dialog("Hello .. are you a student or a lecturer?");
        if ( response == "lecturer" || response == "Lecturer" )
        {
            message ("You look too y ng to be a lecturer");
            message ("I was sure you w re a student!");
        }
        message ("Well I'd better get on …");
        if ( response == "lecturer" || respons == "Lecturer" )
        {
            message (" … with my studies
        }
}
```

2 conditions are combined into one, using OR (||)

&&

(the logical AND operator)

Combining conditions

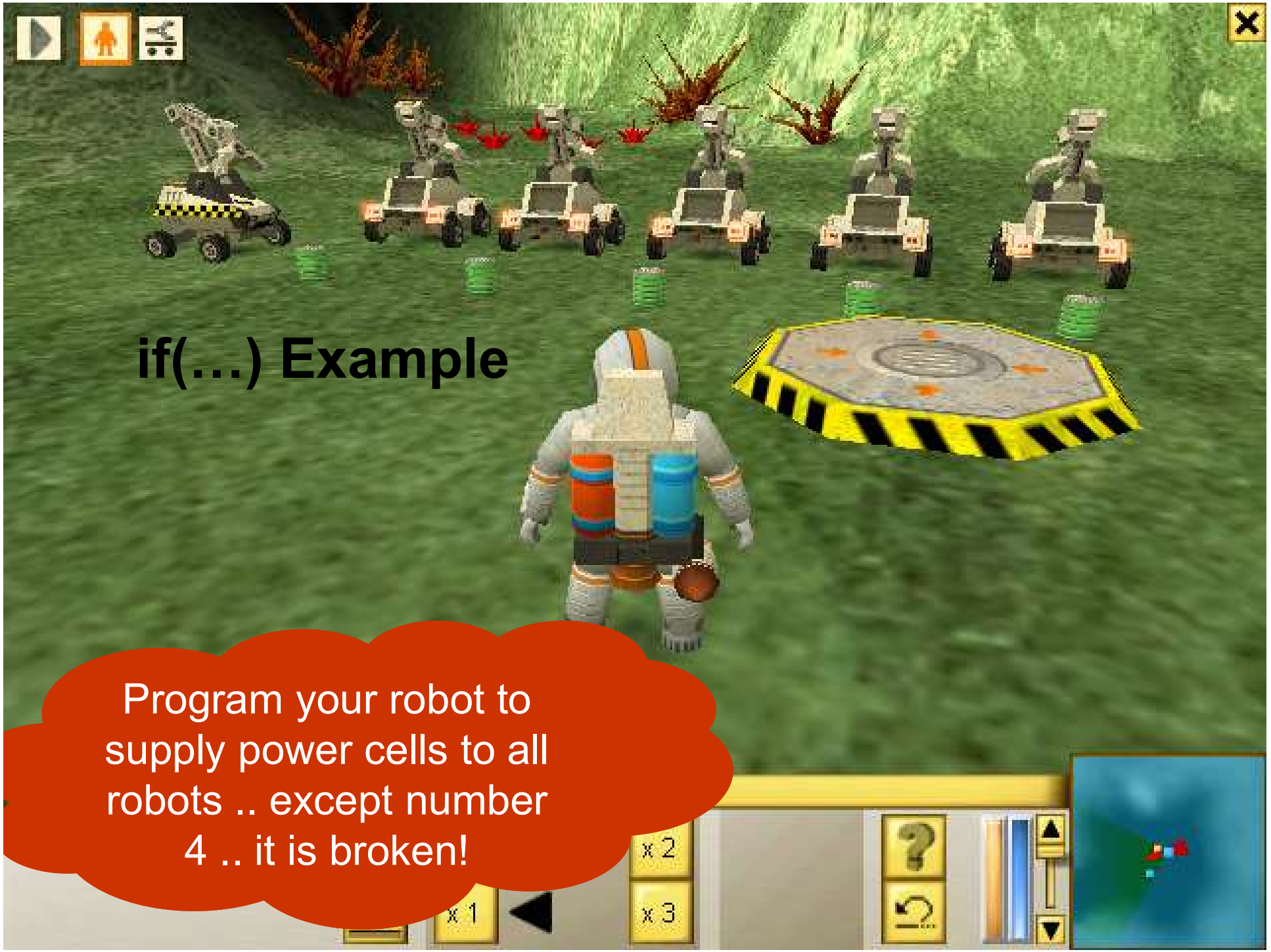
Example using && (AND)

Slag off the lecturers .. More secure version

```
extern void object:: SlagOfflecturerandLecturer()
{
        string response;
        response = dialog("Hello .. are you a student or a lecturer?");
        if ( response != "lecturer" && response != "Lecturer" )
        {
            message ("Thank God .. l turers are so boring and stupid");
            message (" … and so ugly.
        }
        message ("Well I'd better get on …");
        if ( response != "lecturer" && respons != "Lecturer" )
        {
            message (" … down to the ba
        }
}
```

2 conditions are combined into one, using AND (**&&**)

if(…) example


if(…) Example
Program your robot to supply power cells to all robots .. except number 4 .. it is broken!
x 1
x 2
x 3

Power Supplier

Algorithm

1. **Loop** 5 times
   a. Move forward 3 metres
   b. **if** position NOT 4
      i. Turn right and grab cell
      ii. Turn round and install cell
      iii. Turn right, back to path
   end if
End loop

```
extern void object::Supply()
{
    // author: B Ward : 18/12/2010

    for ( int count = 1 ; count <= 5 ; count ++ )
    {
        move (3);
        if ( count != 4 )
        {
            turn(-90); grab();
            turn(180); drop();
            turn(-90);
        }
    }

    message("All Done");
}
```

if(…) else ..

Choose to do one thing or another

Syntax (grammar rules)

The **if(…) else** statement allows 2 alternative choices to be made

```
extern void object:: ifelseProgram()
{
    //   A :  the beginning of program goes here

        if ( condition )
        {
            // B :
            // code here is done once if condition is true.
        }
        else
        {
            // C :
            // code here is done once if condition is false.
        }

    //   D :  this part of the program is done after the if() else statement
    //   so there are 2 paths:  A-B-D (condition true) or A-C-D (false)
}
```

An Example

Chat up anyone

```
extern void object:: Enquire()
{
        string response;
        response = dialog("Hello .. are you a student or a lecturer?");
        if ( response == "lecturer" )
        {
            message ("You look too young to be a lecturer");
            message ("I was sure you were a student!");
        }
        else
        {
            message ("Thank God .. lecturers are so boring and stupid");
            message (" … and so ugly!");
        }
        message ("Well I'd better get on …");
}
```

What are the possible results?

2 possible conversations

The condition true path

Hello .. are you a student or a lecturer?

lecturer

You look too young to be a lecturer I was sure you were a student! W I'd better get on …

OR

The condition false path

Hello .. are you a student or a lecturer?

student

Thank God .. lecturers are so boring and stupid … and so ugly!
Well, I'd better get on …

if(…) else ladders

For multiple choices

An Example of an if() else ladder

Tactful Chatups

```
extern void object:: BeTactful()
{
        string response = dialog("Hello .. what do you do here?");
        if ( response == "lecturer" )
        {
            message ("Oh how nice .. Yawn!");
        }
        else if ( response == "student" )
        {
            message ("Aren't you a bit old for a student?");
        }
        else if ( response == "Director" )
        {
            message ("Can I borrow some money?");
        }
        else
        {
            message ("You're too boring to talk to!");
        }
}
```


What I Learned Today
1. Selection
2. if() statement
3. Conditions
4. Logical operators
|| and &&
5. if() .. else .. statement
6. if() .. else ladders
7. Nested if() statements
THE END

Extra Reading

Nested
if () statements

ifs inside ifs

An Example of a Nested if() statement

Help!

```
extern void object:: GetHelp()
{
        string response1 = dialog("Hello .. what do you do here?");
        if (response1 == "lecturer")
        {
            message ("Of course, sir, I can tell you have a big brain, sir");
            string response2 = dialog("and what do you teach, sir?");
            if ( response2 == "Programming")
            {
                message ("My favourite subject!!");
                message("Perhaps you can help me with this exercise?");
            }
        }
        else
        {
            message ("I'm looking for a programming lecturer!");
        }
}
```

The switch() instruction (for multiple selection)

The switch instruction

General format

```
switch (variable)
{
    case value1:    statement(s);
                    break;
    case value2:    statement(s);
                    break;
    case value3:    statement(s);
                    break;
          ..........................
          ..........................
    default:        statement(s);
}
```

The break statement causes the switch statement to finish here

This is done if no match elsewhere

Example Switch Program

```
extern void object::switchprog()
{
    string input;
    int  numburgers ;                    // number of burgers
    message ("Greed Program") ;
    input = dialog(" How many burgers can you eat? ") ;
    numburgers = strval(input);          // convert to a number

    switch  (numburgers)
    {

        case 1 :   message ( " That's fine " ) ;   break ;
        case 2 :   message ( " A bit excessive " ) ;   break ;
        case 3 :   message ( " Wow, mind your heart! " ) ;   break ;
        default  :   message( " No, I don't believe you!!! " );

    }

    message (" End of program ") ;
}
```

Example

```
extern void object::switchprog2()
{
        string input;   int  month ;          // number of the month
         message ( " The Seasons of the Year " );
         input = dialog ( " Enter the month number (1 TO 12) " );
         month = strval(input);
        switch (month)
        {
                case 11: case 12: case 1: case 2:
                                message ( " It is Winter " );   break;
                case 3: case 4: case 5:
                                message ( " It is Spring " );   break;
                case 6: case 7: case 8:
                                message ( " It is Summer " ); break;
                case 9: case 10:
                                message (" It is Autumn " );   break;
                default:
                                message ( " Error in month number " );
        }
         message ("The End") ;
}
```

default is done for other cases that don't match

The break statement

Give us a break

break is used to break out of a switch statement (or a loop) prematurely
and continue with the rest
of the program.

The continue statement

May I continue?

continue can be used in loop statements to skip over part of the loop and continue to the next repeat

Example

Below is part of a program that performs calculations on a series of 100 numbers. However, continue is used to prevent this for the numbers 25 to 50.

```
for ( n = 1 ; n <= 100 ; n++ )
{
        if ( n >= 25  &&  n <= 50 ) continue ;
        // otherwise do the rest of loop

 }
```