Multimedia Programming 2004

Lecture 2

Erwin M. Bakker
Joachim Rijsdam

Recap

- Learning C++ by example
- No groups: everybody should experience developing and programming in C++!
- Assignments will determine final grade
- Assignments and other materials available on the site: www.liacs.nl/~erwin/MMP2004
- Assignment work should be posted on the Bulletin Board

C++

- ANSI C standard definition (1983-1989)
- C++
- Compiling
  1. C++ code
  2. Compilation
  3. Object code
  4. Linking
  5. Executable

ANSI C Reserved Words

auto, break, case, char, const, continue, default, do, double, else, enum, extern, float, for, goto, if, int, long, register, return, short, signed, sizeof, static, struct, switch, typedef, union, unsigned, void, volatile, while

I/O, Expressions and Statements

- Formatted I/O <stdio.h>
  - printf("Text with format control %s",stringvar)
  - scanf("%d",integervar)
- Output streams <iostream.h>
  #include <iostream.h>
  cout << "Hello World\n";
  cin >> number1 >> number2;
- Expressions and Statements
  - a+b 1 x=p+d
  - 38/5 39%4 (is equal to mod)
  - a = a +1 a += 1 a *=3 a++ ++a a-- --a
- Statements
  - i=3; b = (i++); c = (++i);
  (after this statement: b = 3 and c = 5)

Types and Variables

- char (1 byte)
- short (short int) (2 byte)
- int (2 or 4 bytes)
- enum (2 or 4 bytes)
- long (long int) (4 bytes)
- float (4 bytes)
- double (8 bytes)
- long double (10 bytes)
- unsigned and signed to indicate the use of a sign in the number
- boolean type bool, 0 = false, 1 = true

Types and Variables

- int n=100; declaration and definition
- int n; declaration only
- const int i = 5; // n is not allowed to change
  i = 10; // Error
  i++; // Error
  volatile is the opposite of const but varies amongst systems
- float number;
- enum days { Sunday, Monday,Tuesday, Wednesday, Thursday, Friday, Saturday}
  today;
  today = Monday;

Special Keywords

- register int a

To indicate that a, if possible should be placed in a register (very fast memory) of the CPU, because a is intensively used

Operators

- logical: <, >, <=, >=, ==, !=, &&, ||, !
- arithmetic: +, *, -, /, %, --, ++, sizeof, ,
- bitwise: &, |, ^ (XOR), ~(negation), << (shift left), >> (shift right) (bit-operations)
- +=, -=, *=, /=, %=, &=, |=, ^=, <<=, >>=

Special Statements

Conditional Statements

- if (expression) statement else statement

Iterative Statements

- while (expression) do statement
- do statement while (expression);
- for (expr1; expr2; expr3) statement

Switch Statement

- switch (expression) statement

Break and Continue

- statement: a = x+1;
- compound statement: { a=x+1; b=3; }

while (boolean expression) do
{ ….; break; …; continue;..} next statement;

- break interrupts the loop and continues with next statement;
- continue interrupts the loop and continues with boolean expression

Structuring Your Program Pseudo Code

Calculator Example:

```
while (continue)
do
    read operand1
    read operator
    read operand2
    calculate result
    print result
od
```

Structuring Your Program
Top Down

```
function continu
    return true if continue
    return false if end

function calculate
    return result = operand1 (operator) operand2

function read_operator

main program
    while (continue)
    do
        read operand1        (scanf function from standard lib)
        read operator
        read operand2        (scanf function from standard lib)
        calculate result
        print result  (printf function from standard lib)
    od
```

Functions

- In C only functions
- Parameters by value
- Function definitions never within other functions

```
#include <stdio.h>
#include <math.h>

int main(void)
{
    // Declaration of the function times
    int times(int x, int y);
    cout << times(x,y) << sin(times(x,y)) << '\n';
    return 0;
}

// Definition of the function times
int times(int x; int y)
{
    int result;
    result = x*y;
    return result;
}
```

Functions

- void print_header(void)

No parameters, no results

- int main(void)

No parameters, result an int

- int smallest(int x, int y, int z)

Three integer parameters and result int

Functions Recursion

```
int power(int x, int exponent)
{
        if (exponent == 0)
        {
                return 1;
        }
        else
        {
                return x * power( x, exponent – 1 );
        }
}
```

Functions: Parameters by Value

```
void swap(int x, int y)
{
        int temp;
        temp = y;
        y = x;
        x = temp;
}

void main(void)
{ int a =10, b = 11; swap(a,b); }
```

- Parameters are given by value: x and y can be seen as local variables that are initialized at the call of the function
- After swap(a,b) executed, the variables a and b passed through the parameters will still have their original values, as only their respective values have been passed to the local int x and int y!