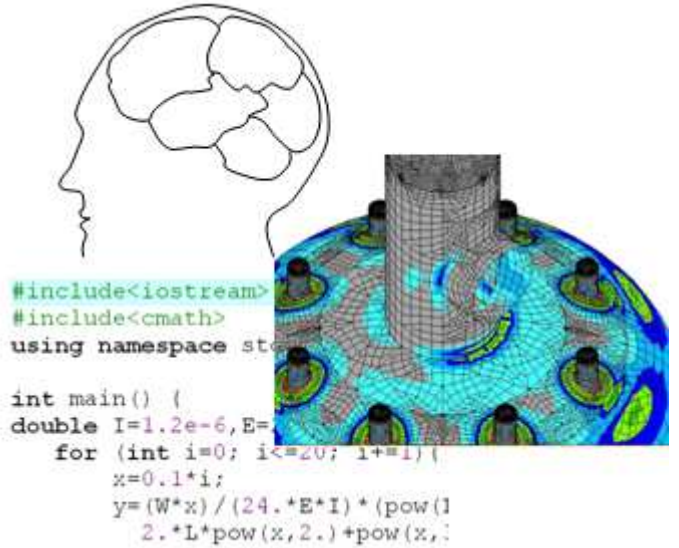
ME 240
Computation for
Mechanical
Engineering



Lecture 5

C++ operators, intrinsic functions, and strings

Contents

Extracted from `http://cpp.gantep.edu.tr`

- **Basic Operators**
- **Basic Strings**
- **Header Files**
- **Basic Intrinsic Functions**

Basic Operators

Operators are special symbols that perform operations with variables and constants.

Arithmetic operators

Operator	Description	Example	Result
+	Addition	13 + 5	18
–	Subtraction	13 – 5	8
*	Multiplication	13 * 5	65
/	Division	13 / 5	2
%	Modulus (remainder of x/y)	13 % 5	3

Operator precedence: first **( )**, then ***** and **/** , and finally **+** and **–**

```
2 - 3 * 4 + 2 = -8
2 * 3 + 4 - 2 = 8
2 * (3 + 4) - 2 = 12
3 * 5 / 3 = 5
10 / 2 * 3 = 15
(5 + (11-5) * 2) * 4 + 9 = 77
```

evaluate left-to-right!

The assignment operator (=)

```
int x, y;
x = 2;
y = 5*x;   // y = 10
x = x + 4; // x = 6
y = y/2;   // y = 5
```

Chained assignment

```
m = (n = 66) + 9; // n = 66 and m = 75
x = y = 22;       // x = 22 and y = 22
```

Compound assignment operators (+=, -=, *=, /=, %=)

Operator	Description	Example	Equivalent to
+=	add and assign	`x += 3`	`x = x + 3`
-=	subtract and assign	`x -= 5`	`x = x - 5`
*=	multiply and assign	`x *= 4`	`x = x * 4`
/=	divide and assign	`x /= 2`	`x = x / 2`
%=	find reminder and and assign	`x %= 9`	`x = x % 9`

Note that **`x *= a+b`** expands to **`x = x * (a+b)`**

which is generally not the same as **`x = x * a+b`**

Similarly **`x /= a+b`** expands to **`x = x / (a+b)`**

Increase and decrease by 1 (++, --)

- The following are equivalent in functionality

```
x = x + 1;
x += 1;
x++;
```

```
x = x - 1;
x -= 1;
x--;
```

- ++ and -- can be used both as a prefix and as a suffix.

```
a = 5;
b = a++;
```
assigns **b=5** and then **a=6**

```
a = 5;
b = ++a;
```
assigns **a=6** and then **b=6**

Integer division

```
int i, j, k;
double p, q;
i = 4/2;      results in the assignment i=2
j = 5/2;      results in the assignment j=2
p = 5/2;      results in the assignment p=2.0
p = 5/2.0;    results in the assignment p=2.5
q = i + p;    results in the assignment q=2.0+2.5 = 4.5;
k = 25.0/2;   results in the assignment k=12
```

Type casting

```
int i;
double d;
i = int(7.25);  results in the assignment i=7
d = double(5);  results in the assignment d=5.0
```

The `sizeof()` operator

The operator **`sizeof()`** is used to calculate the size in bytes of data types, variables, arrays or literals.

For example

```
int i;
double d;
cout << "sizeof(int)   = " << sizeof(int)   << " bytes" << endl;
cout << "sizeof(float) = " << sizeof(float) << " bytes" << endl;
cout << "sizeof(double)= " << sizeof(double)<< " bytes" << endl;
cout << "sizeof(i)     = " << sizeof(i)     << " bytes" << endl;
cout << "sizeof(d)     = " << sizeof(d)     << " bytes" << endl;
```

Output

```
sizeof(int)   = 4 bytes
sizeof(float) = 4 bytes
sizeof(double)= 8 bytes
sizeof(i)     = 4 bytes
sizeof(d)     = 8 bytes
```

Basic Strings

- A *string* is a series of characters, such as "Hello World!"

- A string variable can be declared and assigned as follows:

```
string s = "This is string";
```

Note that you need to include the **<string>** header.

- Some basic operations can be performed on strings.

```
string s1, s2, s3, s4;
s1 = "centi";
s2 = "meter";
s3 = s1;          results in the assignment  s3="centi"
s4 = s1 + s2;     results in the assignment  s4="centimeter"
```

Example: *Using strings*

```
#include <iostream>
#include <string>
using namespace std;

int main () {

  string name;

  cout << "What is your name? ";
  cin >> name;
  cout << "Hello " << name << endl;

}
```

Output

```
What is your name? Mert
Hello Mert
```

Header Files

- The **`#include`** directive allows the program to use source code from another file.
- **`#include <iostream>`**

  refers to an external file named **`iostream`**, and tells the preprocessor to take the **`iostream`** file and insert in the current program.
- The files that are *included* are called *header files*.
- The C/C++ standard library traditionally declares standard functions and constants in header files.

C++ Standard Library	Standard Template Library	C Standard Library
ios	vector	cassert
iostream	deque	cctype
iomanip	list	cerrno
fstream	map	climits
sstream	set	clocale
	stack	cmath
	queue	csetjmp
	bitset	csignal
	algorithm	cstdarg
	functional	cstddef
	iterator	cstdio
		cstdint
		cstdlib
		cstring
		ctime

Basic Intrinsic Functions

An *intrinsic* or a *library* function is a function provided by the C++ language.

For example the **`cmath`** library contains mathematical functions/constants:

Some C++ library mathematical functions and constants defined in <cmath>

Function Decleration	Description	Example	Result
`double fabs(double x);`	absolute value of real number, |x|	`fabs(-4.0)`	4.0
`int floor(double x);`	round down to an integer	`floor(-2.7)`	-3
`int ceil(double x);`	round up to an integer	`ceil(-2.7)`	-2
`double sqrt(double x);`	square root of x	`sqrt(4.0)`	2.0
`double pow(double x, double y);`	the value of x^y	`pow(2., 3.)`	8.0
`double exp(double x);`	the value of e^x	`exp(2.0)`	7.38906
`double log(double x);`	natural logarithm, $\log_e x = \ln x$	`log(4.0)`	1.386294
`double log10(double x);`	base 10 logarithm, $\log_{10} x = \log x$	`log10(4.0)`	0.602060
`double sin(double x);`	sinus of x (x is in radian)	`sin(3.14)`	0.001593
`double cos(double x);`	cosine of x (x is in radian)	`cos(3.14)`	-0.999999
`double tan(double x);`	tangent of x (x is in radian)	`tan(3.14)`	-0.001593
`double asin(double x);`	arc-sine of x in the range [-pi/2, pi/2]	`asin(0.5)`	0.523599
`double acos(double x);`	arc-cosine of x in the range [-pi/2, pi/2]	`acos(0.5)`	1.047198
`double atan(double x);`	arc-tangent of x in the range [-pi/2, pi/2]	`atan(0.5)`	0.463648
`M_PI`	constant pi	`myPI = M_PI`	3.141592...
`M_E`	constant e	`x = M_E`	2.718281...

Some standard C++ library functions and constant defined in <cstdlib>

Function Decleration	Description	Example	Result
`int abs(int x);`	absolute value of integer number, |x|	`abs(-4);`	4
`int atoi(const char *s);`	converts string to integer	`atoi("-1234")`	-1234
`double atof(const char *s);`	converts a string to double	`atof("123.54")`	123.54
`void exit(int status);`	terminates the calling process "immediately"	`exit(1);`	-
`int rand(void);`	Returns a random integer between 0 and RAND_MAX	`rand()`	1048513214
`RAND_MAX`	The largest number rand() will return	`x = RAND_MAX`	2147483647

Example program: *Using trigonometric functions*

```
#include <iostream>
#include <cmath>
using namespace std;

int main () {

  double beta;
  cout << "Input an angle in degrees: ";
  cin >> beta;

  beta = beta * M_PI/180.0; // convert to radians
  cout << "sin(beta) = " << sin(beta) << endl;
  cout << "cos(beta) = " << cos(beta) << endl;
  cout << "tan(beta) = " << tan(beta) << endl;

}
```

Output

```
Input an angle in degrees: 60
sin(beta) = 0.866025
cos(beta) = 0.5
tan(beta) = 1.73205
```

Example program: *Using logarithmic functions*

```
#include <iostream>
#include <cmath>
using namespace std;

int main (){

  double x;
  cout << “Input a value: ";
  cin >> x;
  cout << "log(x)    = " << log(x)      << endl;
  cout << "log10(x)  = " << log10(x)    << endl;
  cout << "exp(x)    = " << exp(x)      << endl;,
  cout << "pow(x,2.5)= " << pow(x,2.5) << endl;

}
```

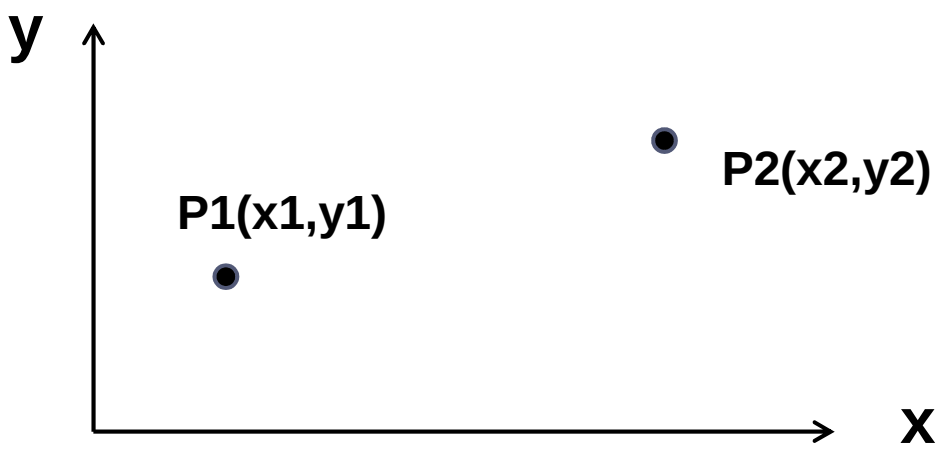
Output

```
Input a value: 1.4
log(x)    = 0.336472
log10(x)  = 0.146128
exp(x)    = 4.0552
pow(x,2.5)= 2.3191
```

Example

- Write a program to find the distance between two poins. Hint:

$$\text{Distance} = \sqrt{(x2-x1)^2 + (y2-y1)^2}$$



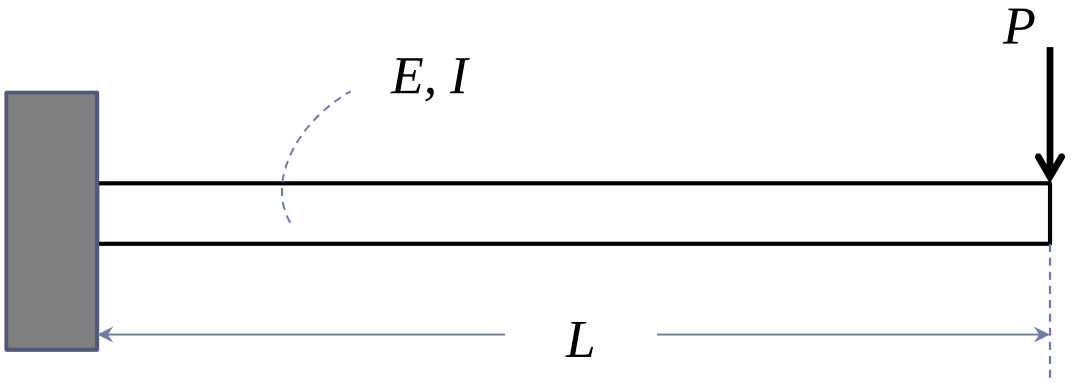
```
#include <iostream>
#include <cmath>
using namespace std;

int main () {
  double x1,x2,y1,y2,distance;
  cout<<"input the coordinates of first point"<<endl;
  cin>>x1>>y1;
  cout<<"input the coordinates of second point"<<endl;
  cin>>x2>>y2;
  distance=sqrt(pow(x2-x1,2.)+pow(y2-y1,2.));
  cout<<"the distance is "<<distance<<endl;
system("pause");
}
```

Example:

- Find the deflection of a cantilever beam under the action of an end load as shown in Figure. Hint:

$$\text{Deflection} = \frac{P * L}{3 * E * I}$$



```
#include <iostream>
#include <cmath>
using namespace std;
int main () {
    double P,l,E,In,deflection;
    cout<<"input the load\n";
    cin>>P;
    cout<<"input the length of beam\n";
    cin>>l;
    cout<<"input the inertia\n";
    cin>>In;
    cout<<"input the Young's modulus\n";
    cin>>E;
    deflection=P*pow(l,3.)/(3*E*In);
    cout<<"deflection is "<<deflection<<endl;
system("pause");
}
```