Introduction to C++

"Programming is the activity of communicating algorithms to computers." Kelly & Pohl.

C++ Keywords

The following *keywords* are part of the standard C++ language. They should not be used as identifiers for variables or functions. Keywords particular to C++ are shown in boldface.

asm	auto	break	case	**catch**
char	**class**	const	continue	default
delete	do	double	else	enum
extern	float	for	**friend**	goto
if	**inline**	int	long	**new**
operator	**private**	**protected**	**public**	register
return	short	signed	sizeof	static
struct	switch	**template**	**this**	**throw**
try	union	unsigned	**virtual**	void
volatile	while			

1 Program Organization in C++

In a *procedural language* a program consists of static data and a collection of functions.

```
// Global data

int checking_balance, credit_balance, credit_limit;

// main procedure (function)

void main ()
{
    checking_balance = 1000;
    credit_balance = 500;
    credit_limit = 2000;

    int cost = 500;

    purchase (cost);
}

void purchase (int cost)
{
    if (cost < checking_balance)
      pay_by_cash (cost);
    else
      pay_by_credit (cost);
}

void pay_by_cash (int cost)
{
    checking_balance -= cost;
    if (checking_balance == 0) cout << "You're broke!";
    if (checking_balance < 0)
      transfer_from_savings (abs(checking_balance));
}

void pay_by_credit (int cost)
{
    if (credit_balance + cost > credit_limit)
      cout << "You've exceeded your credit limit!";
    else
```

```
      credit_balance += cost;
}
```

C++ supports procedural programming. It also allows another more powerful form of organization known as *object oriented programming*.

```
// Classes
class CreditCard {
private:
    int balance;
    int limit;
public:
    void purchase (int cost);
};

class CheckingAccount {
private:
    int balance;
public:
    void purchase (int cost);
};

class Consumer {
private:
    CreditCard plastic;
    CheckingAccount checking;
public:
    Cosumer (int starting_balance, int credit_limit);
    void purchase (int cost);
    void pay_by_cash (int cost);
    void pay_by_credit (int cost);
};

void main ()
{
    Consumer Tom(100, 1000), Andrea(250, 5000);
```

```
    Tom.purchase (500);
    Andrea.purchase (120);
}
```

Each class is designed to represent a single concept. An instance of a class is called an *object*.

Intrinsic Types

Basic data types provided by C++ are

char	character
int	integer
long	long integer
float	single precision floating point
double	double precision floating point

The **char**, **int**, and **long** types may be **signed** or **unsigned**. The intrinsic types may be mixed in assignment expressions, *e.g.*

```
int i = 10;
unsigned long l;

l = i;
```

The **sizeof()** operator may be used to determine the size in bytes of any type.

Operators

C++, like C, provides a rich set of operators which fall under the general categories

- Relational Operators

- Arithmetic Operators
- Bitwise Operators
- Logic Operators
- Assignment Operators
- Reference and Dereference Operators

The relational operators are

```
==      equal to
!=      not equal to
<       less than
>       greater than
<=      less than or equal to
>=      greater than or equal to
```

The arithmetic operators are

```
+       add
-       subtract
*       multiply
/       divide
%       remainder
++      increment
--      decrement
```

The bitwise operators are

```
&       bitwise AND
|       bitwise OR
~       bitwise complement
^       bitwise XOR
<<      bitwise left shift
>>      bitwise right shift
```

The logic operators are

```
&&      AND
||      OR
!       NOT
```

The assignment operators are

```
=       assignment
+=      arithmetic assignment
-=
*=
/=
%=
<<=     bitwise assignment
>>=
&=
|=
^=
```

Flow Control Structures

C++ provides flow control structures for branching and looping like other structured languages. These are

```
if () {} else {}
if () {} else if () {} ... else {}
switch () { case 0: {} case 1: {} ... default: {}}
goto
while () {}
do {} while ()
```

The `break` and `continue` statements also control branching within loops.

Notes

- All variables must be declared.
- All functions must be declared.
- A variable declared inside a function is local to that function.
- Anything declared outside a function is visible to (may be referenced) by everything following the declaration.

- Arrays have a starting index of 0.
- Strings are arrays of type **char** with the last character being the *null* character.
- Arguments to functions are passed by value in C. C++ supports passing by reference.

Functions in the C Standard Math Library

Be sure to include the math header file in any program that uses the math library functions.

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

Most of the functions take a double value as their argument, and return a double value. They would be declared as

```
double func (double x);
```

Name	**Description**
acos	Return arc cosine of x
asin	Return arc sine of x
atan	Return arc tangent of x
atan2	Requires arc tangent of x/y
atof	Convert string to floating point number
ceil	find smallest integer not less than x
cos	Return cosine of x (radians)
cosh	Return hyperbolic cosine of x
exp	Return e^x
fabs	Return absolute value of x
floor	Return largest integer not greater than x
fmod	Return remainder of x/y
frexp	Split argument into mantissa and exponent
ldexp	Return value given mantissa and exponent
log	Return natural logarithm of x
log10	Return base 10 log of x
modf	Break a number into whole and fractional parts
pow	Return x^y
sin	Return sine of x (radians)
sinh	Return hyperbolic sine of x
sqrt	Return positive square root of x
tan	Return tangent of x (radians)
tanh	Return hyperbolic tangent of x