Υπερφόρτωση Τελεστών

Υπερφόρτωση Τελεστών

- Τύποι Τελεστών
  - Built in (**`int`**, **`char`**) ή ορισμένοι από τον χρήστη (user-defined)
  - Χρήση υπαρχόντων τελεστών με τύπους ορισμένους από τον χρήστη
- Υπερφόρτωση Τελεστών
  - Δημιουργία συνάρτησης για την κλάση
  - Το όνομα της συνάρτησης είναι '**`operator`**' ακολουθούμενο από το σύμβολο του τελεστή Π.χ:
    - **`Operator+`** για τον τελεστή πρόσθεσης **+**

Υπερφόρτωση Τελεστών σε Κλάση

- Χρήση τελεστών σε αντικείμενα κλάσεων
  - Πρέπει πρώτα να γίνει υπερφόρτωση τους για την κλάση
    - Εξαιρέσεις (δεν χρειάζεται υπερφόρτωση):

      Τελεστής ανάθεσης, `=`

      Memberwise ανάθεση μεταξύ αντικειμένων

      Τελεστής διεύθυνσης, `&`

      Επιστρέφει τη διεύθυνση του αντικειμένου
    - Μπορούν να υπερφορτωθούν αν θέλουμε διαφορετική συμπεριφορά (όπως και ο τελεστής κόμμα: ,)
  - Προσφέρουν πιο σύντομη και κατανοητή διατύπωση:

```
object2 = object1.add(object2);
object2 = object2 + object1;
```

Διαθέσιμοι Τελεστές

Τελεστές που μπορούν να υπερφορτωθούν							
+	-	*	/	%	^	&	|
~	!	=	<	>	+=	-=	*=
/=	%=	^=	&=	|=	<<	>>	>>=
<<=	==	!=	<=	>=	&&	||	++
--	->*	,	->	[]	()	new	delete
new[]	delete[]						

Τελεστές που δεν μπορούν να υπερφορτωθούν				
.	.*	::	?:	sizeof

Υπερφόρτωση Τελεστή ως μέλος κλάσης ή ως Friend συνάρτηση

- Operator functions
  - Ως μέθοδος (μέλος της κλάσης)
    - Το αριστερό μέλος του τελεστή είναι το αντικείμενο για το οποίο καλείται (μπορούμε να χρησιμοποιήσουμε και τον τελεστή **this** για να αναφερθούμε σε αυτό). Προφανώς είναι ιδίου τύπου με την κλάση στην οποία ορίζουμε τον τελεστή.
    - Το δεξιά μέλος του τελεστή είναι όρισμα της συνάρτησης (οτιδήποτε τύπου θέλουμε)
  - Ως συνάρτηση (εκτός της κλάσης, όχι μέθοδος)
    - Πρέπει να δοθούν και οι δύο παράμετροι (πριν και μετά τον τελεστή) ως ορίσματα
    - Πρέπει να είναι friend συνάρτηση (προσθέτουμε σχετική δήλωση στην κλάση) για να προσπελάσει **private** ή **protected** μέλη
  - (), [], ->, = πρέπει να οριστούν ως μέλη της κλάσης
  - Καλούνται
    - Όταν το αριστερό μέλος του τελεστή (για binary τελεστές) είναι αντικείμενο της κλάσης
    - Το μοναδικό όρισμα (για unary τελεστές) είναι αντικείμενο της κλάσης

Παράδειγμα

```
class myDate{
  friend ostream &operator<<( ostream &, const myDate & );
  friend istream &operator>>( istream &, myDate & );

  public: myDate(int d=5, int m=5, int y=1994) :day(d), month(m), year(y) {}
    bool operator==( const myDate &right ) const;
    myDate& operator++(); myDate operator++( int );
    int operator[]( char i );

  private:
    int day, month, year;
};
```

Υπερφόρτωση συναρτήσεων μη- μελών της κλάσης (απλά δήλωση ως friend)

Υπερφόρτωση συναρτήσεων μελών της κλάσης

prefix/postfix increment operator
Dummy parameter “int”

Παράδειγμα

```
class myDate{
  friend ostream &operator<<( ostream &, const myDate & );
  friend istream &operator>>( istream &, myDate & );
  public: myDate(int d=5, int m=5, int y=1994) :day(d), month(m), year(y) {}
    bool operator==( const myDate &right ) const;
    myDate& operator++(); myDate operator++( int );
    int operator[]( char i );
  private:
    int day, month, year;
};
```

```
bool myDate::operator==( const myDate &right ) const{
  if ((day == right.day) && (month == right.month) && (year == right.year))
    return true;
  return false;
}

int myDate::operator[]( char i ){
  switch (i){
    case 'm':
      return month;
    case 'y':
      return year;
    case 'd':
      return day;
    default:
      return -1;
  }
}
```

Παράδειγμα

```
class myDate{
  friend ostream &operator<<( ostream &, const myDate & );
  friend istream &operator>>( istream &, myDate & );
  public: myDate(int d=5, int m=5, int y=1994) :day(d), month(m), year(y) {}
    bool operator==( const myDate &right ) const;
    myDate& operator++(); myDate operator++( int );
    int operator[]( char i );
  private:
    int day, month, year;
};
```

```
istream &operator>>( istream &input, myDate &a ){
  input >> a.year >> a.month >> a.day;
  return input;
}

ostream &operator<<( ostream &output, const myDate &a ){
  output << a.year << "/" << a.month << "/" << a.day;
  return output;
}

myDate &myDate::operator++() {
  day++; return *this;
}

myDate myDate::operator++( int ) {
  myDate temp = *this;
  day++;
  return temp;
}
```

Παράδειγμα

```
class myDate{
  friend ostream &operator<<( ostream &, const myDate & );
  friend istream &operator>>( istream &, myDate & );
  public: myDate(int d=5, int m=5, int y=1994) :day(d), month(m), year(y) {}
    bool operator==( const myDate &right ) const;
    myDate& operator++(); myDate operator++( int );
    int operator[]( char i );
  private:
    int day, month, year;
};
```

```
int main(){
  myDate dt1(15,7,2015);
  myDate dt2(25,9,2014);
  myDate dt3;
  cin >> dt3;
  cout << dt2 <<endl;
  cout << ++dt1; cout << dt1++ <<endl;
  cout << "Month: " << dt2['m'] <<endl;
  cout << ((dt1 == dt2)?"same date":"different date");
}
```

Πρόσθετο Υλικό

- Μελετήστε και τα παραδείγματα από το **Κεφάλαιο 10** του βιβλίου:
  «C++ How to Program, 9/e Paul & Harvey Deitel»
  http://media.pearsoncmg.com/ph/esm/deitel/cpp_htp_9/code_examples/Code_Examples.zip