Preporučeni načini preklapanja operatora

- Od verzije C++20 postoji operator <=> koji se može preklopiti, a kad je on preklopljen, implicitno su definisani i svi drugi relacioni operatori u prirodnom obliku:

```
int operator<=> (const string& s1, const string& s2) {
  return strcmp(s1.str,s2.str);
}
```

- Kad je ovaj operator preklopljen, implicitno je definisano i svih šest ostalih relacionih operatora u prirodnom obliku:

```
inline bool operator== (const string& s1, const string& s2) { return (s1<=>s2)==0; }
inline bool operator!= (const string& s1, const string& s2) { return !(s1==s2); }
inline bool operator< (const string& s1, const string& s2) { return (s1<=>s2) < 0; }
inline bool operator> (const string& s1, const string& s2) { return s2<s1; }
inline bool operator<= (const string& s1, const string& s2)
  { return (s1<s2)||(s1==s2); }
inline bool operator>= (const string& s1, const string& s2) { return (s2<=s1); }
```

- Može se tražiti i generisanje podrazumevane implementacije ovog operatora, koja se svodi na isti operator primenjen redom na podobjekte, u poretku njihovog deklarisanja:

```
bool operator<=> (const X&, const X&) = default;
```

Preporučeni načini preklapanja operatora

- Kada se preklapaju operatori << i >> za klase sa ciljem implementacije izlaznih odnosno ulaznih operacija preko znakovnih tokova (*I/O stream*), moraju se definisati kao funkcije nečlanice, jer im je drugi (desni) operand onaj za koji se definišu
- Oni treba da vraćaju referencu na isti objekat koji je dostavljen kao prvi operand:

```
std::ostream& operator<< (std::ostream& os, const T& object) {

  // Write object to output stream os

  return os;
}

std::istream& operator>> (std::istream& is, T& object) {

  // Read object from input stream is

  if (/* T could not be created */ )
    is.setstate(std::ios::failbit);

  return is;
}
```