```
#ifndef OPERAND_H
#define OPERAND_H

class operand{
public:
    int p; char o;
    operand(){}
    operand(const int pre, const char opd):
        p(pre), o(opd){}
    operand& operator()(const int pre, const char opd){
        p=pre; o=opd;
        return *this;
    }
    bool operator<(const operand& opd){
        return p<opd.p;
    }
    bool operator>(const operand& opd){
        return p>opd.p;
    }
    bool operator<=(const operand& opd){
        return p<=opd.p;
    }
    bool operator>=(const operand& opd){
        return p>=opd.p;
    }
    bool operator&&(const operand& opd){
        return p && opd.p;
    }
    bool operator&&(const int& opd){
        return p && opd;
    }
    bool operator||(const int& opd){
        return p || opd;
    }
    bool operator||(const operand& opd){
        return p||opd.p;
    }
    bool operator!(){
        return !p;
    }
    bool operator==(const operand& opd){
        return p==opd.p;
    }
    bool operator!=(const operand& opd){
        return p!=opd.p;
    }
    bool operator==(const int& m){
        return p==m;
    }
    bool operator!=(const int& m){
        return p!=m;
    }
    bool operator>(const int& m){
        return p>m;
    }
    bool operator<(const int& m){
        return p<m;
    }
    bool operator<=(const int& m){
        return p<=m;
    }
    bool operator>=(const int& m){
        return p>=m;
    }
    operand& operator=(const operand& opd){
        p=opd.p;
        o=opd.o;
        return *this;
```

```
69      }
70
71  };
72  #endif
```