13.8 — Overloading the increment and decrement operators

ALEX JULY 31, 2021

Nt cpjm_bg eef cegl apck cl re (`++`) ę l be bcapck cl re (`--`) emncp_rmpqepnpcrrw qrp_ gef rdmpu _pbl u grf ml cek _jjevacnrgml Gf cpce pce ars_jjwu mt cpqgml q nd f c gl apck cl re l be bcapck cl re mncp_rmpq enpcdy gl apck cl re l be bcapck cl re (`++x;` `--y;`) ę l be nmqrdy gl apck cl re l be bcapck cl re (`x++;` `y--;`) .

Aca_s qc f cegl apck cl re l be bcapck cl re mncp_rmpq pce`mrf s l _pw mncp_rmpq l be f cw k mbg w f cg mncp_l bq f cw bce`cqre t cpjm_bcbe qk ck `cped l argml q V clj _ai jc rf c npcdy t cpqgml q e pqre`ca_s qc f cw bc f ck mqreqrp_ gef rdmpu _pb.

Overloading prefix increment and decrement

Opcdy gl apck cl re l be bcapck cl re pc nt cpjm_bcbe v_arjw f ce _k c e q l w mpk _j sl _pw mncp_rmp V clj bm f gql ce`w v_k njcIJ

```cpp
#include <iostream>

class Digit
{
private:
    int m_digit;
public:
    Digit(int digit=0)
        : m_digit{digit}
    {
    }

    Digit& operator++();
    Digit& operator--();

    friend std::ostream& operator<< (std::ostream& out, const Digit&
d);
};

Digit& Digit::operator++()
{
    // If our number is already at 9, wrap around to 0
    if (m_digit == 9)
        m_digit = 0;
    // otherwise just increment to next number
    else
        ++m_digit;

    return *this;
}

Digit& Digit::operator--()
{
    // If our number is already at 0, wrap around to 9
    if (m_digit == 0)
        m_digit = 9;
    // otherwise just decrement to next number
    else
        --m_digit;

    return *this;
}

std::ostream& operator<< (std::ostream& out, const Digit& d)
{
 out << d.m_digit;
 return out;
}

int main()
{
    Digit digit(8);

    std::cout << digit;
    std::cout << ++digit;
    std::cout << ++digit;
    std::cout << --digit;
    std::cout << --digit;

    return 0;
}
```

Ns pęCgegręaj_ qqęf njbqęę sk `cpę`cru ccl ęĩę l bę ħV cŅcęntcpjm_bcbęj apck cl rę l bębcapck cl ręqnęf cwęj apck cl rħbcapck cl ręf cębgegrĤ u p_nnqj eę pns l bęqęf cębgegręęj apck cl rcbħbcapck cl rcbęnsrę_l ecĦ

[illegible]

```
89098
```

[illegible]

Overloading postfix increment and decrement

[illegible]

[illegible]

[illegible]

```
class Digit
{
private:
    int m_digit;
public:
    Digit(int digit=0)
        : m_digit{digit}
    {
    }

    Digit& operator++(); // prefix has no parameter
    Digit& operator--(); // prefix has no parameter

    Digit operator++(int); // postfix has an int parameter
    Digit operator--(int); // postfix has an int parameter

    friend std::ostream& operator<< (std::ostream& out, const Digit&
d);
};

// No parameter means this is prefix operator++
Digit& Digit::operator++()
{
    // If our number is already at 9, wrap around to 0
    if (m_digit == 9)
        m_digit = 0;
    // otherwise just increment to next number
    else
        ++m_digit;

    return *this;
}

// No parameter means this is prefix operator--
Digit& Digit::operator--()
{
    // If our number is already at 0, wrap around to 9
    if (m_digit == 0)
        m_digit = 9;
    // otherwise just decrement to next number
    else
        --m_digit;
```

```
    return *this;
}

// int parameter means this is postfix operator++
Digit Digit::operator++(int)
{
    // Create a temporary variable with our current digit
    Digit temp{*this};

    // Use prefix operator to increment this digit
    ++(*this); // apply operator

    // return temporary result
    return temp; // return saved state
}

// int parameter means this is postfix operator--
Digit Digit::operator--(int)
{
    // Create a temporary variable with our current digit
    Digit temp{*this};

    // Use prefix operator to decrement this digit
    --(*this); // apply operator

    // return temporary result
    return temp; // return saved state
}

std::ostream& operator<< (std::ostream& out, const Digit& d)
{
 out << d.m_digit;
 return out;
}

int main()
{
    Digit digit(5);

    std::cout << digit;
    std::cout << ++digit; // calls Digit::operator++();
    std::cout << digit++; // calls Digit::operator++(int);
    std::cout << digit;
    std::cout << --digit; // calls Digit::operator--();
    std::cout << digit--; // calls Digit::operator--(int);
    std::cout << digit;

    return 0;
}
```

Sf ggenpg rq

```
5667665
```

Sf cpcę pcę ętu ęg rcpcqrg eęf g eqęmg eęml ę cpcH Egpqrĥ mrcęf _ręu cN cębgqrg es gqf cbęf cęnpcdyęębmk ęf cęnmqrdyęmcp_rmpqę`w npmtgbg eę l ęg rcecpębs k k wę_p_k crcpęml ęf cęnmqrdyę cpqgl H Rcaml bĥ ca_s qcęf cębs k k wę_p_k crcpęgę mręqcbęg ęf cębs l argml gk njck cl r_rgml ĥu cęf _tcę mręt cl ęg cl ęręę _k cH Sf gęcjjqęf cęmk ngjcpęmępc_ręf gęt_pg `jcęqęęnj_acf mjbcpĥu f gaf ęk c_l qęgr u ml ęu _pl ęs qęf _ręu cębcaj_pcbęę_pg `jcę s rę ct cpęs qcbęgH

Sf gębĥ mrcęf _ręf cęnpcdyę l bęnmqrdyęmcp_rmpqębmęf cęq_k cęm`ĥęf cwę mrf ęg apck cl ręmpębcapck cl ręf cęm`hcarH Sf cębgdcpcl ac `cru ccl ęf cęu mgęg ęf cęt_js cęf cwęcrs pl H Sf cęntcpjm_bcbęnpcdyęmcp_rmpqęcrs pl ęf cęm`hcarę drcpęgęf _qę`ccl ęg apck cl rcbęmp bcapck cl rcbH Bml qcos cl rjwĥntcpjm_bg eęf cqcęgęęgjwęrp_gęf rdmpu _pbH V cęgk njwęg apck cl ręmpębcapck cl ręms pęk ck `cpęt_pg `jcqĥ _l bęf cl ęcrs pl ęGrf gęH

Sf cęnmqrdyęmcp_rmpqĥml ęf cęmrf cpęf _l bĥ ccbęmęcrs pl ęf cęqr_rcęmdęf cęm`hcarę *cdmpc* ęg apck cl rcbęmpębcapck cl rcbH Sf gęc_bqęm _ęgręmdę ęml s l bps k ĥęgęu cęg apck cl ręmpębcapck cl ręf cęm`hcarĥu cęu ml ęę cę `jcęmęcrs pl ęf cęqr_rcęmdęf cęm`hcarę cdmpcęgręu _q g apck cl rcbęmpębcapck cl rcbH Nl ęf cęmrf cpęf _l bĥęgęu cęcrs pl ęf cęqr_rcęmdęf cęm`hcarę cdmpcęu cęg apck cl ręmpębcapck cl regrĥęf c g apck cl ręmpębcapck cl ręu gjję ct cpę`cę_jjcbH

Sf cęwnga_jęu _węf gęnpm`jck ęgęqmjtcbęgęrmęs qcęęrck nmp_pwęt_pg `jcęf _ręf mjbqęf cęt_js cęmdęf cęm`hcarę cdmpcęgręgęg apck cl rcbęmp bcapck cl rcbH Sf cl ęf cęm`hcareęgqcjdęa_l ę cęg apck cl rcbęmpębcapck cl rcbH @l bęg _jjwęf cęrck nmp_pwęt_pg `jcęgęcrs pl cbęmęf cęa_jjcpH H ęf gęu _wĥęf cęa_jjcpęcacgtcqęęamnwęmdęf cęm`hcarę cdmpcęgręu _qęg apck cl rcbęmpębcapck cl rcbĥ `s ręf cęm`hcareęgqcjdęgęg apck cl rcbęmp bcapck cl rcbH Mmrcęf _ręf gęk c_l qęf cęcrs pl ęt_js cęmdęf cęntcpjm_bcbęmcp_rmpęk s qręę cę ęml ĥpcdcpcl acĥ `ca_s qcęu cęa_l ęcrs pl ę pcdcpcl acęrmęęjma_jęt_pg `jcęf _ręu gjję cębcqrpmwcbęu f cl ęf cęębs l argml ęcvgrqH @jqmę mrcęf _ręf gęk c_l qęf cęnmqrdyęmcp_rmpqę pc

rwngа_jjwąjcqqęcdtgąglrąf_lęfcęnpcdtyęmcp_rmpqę`ca_sqcęmdęfcę bbcbęntcpf c_bęmdęjqr_lrg_rgeeę ęck nmp_pwęt_pg_ `jcęl beępcrsplgeę`w t_jsceęjqrc_bęmdępcdpclacĦ

Egj_jjwĥę mrcęf_ręcŅceęu pgrclęfcęnmqrfg apck clręl beęnmqrĥbcapck clręgeps afęęu_węf_rąrg_jjqęfcęnpcĥg apck clręl beęnpcĥ bcapck clręmbmęk mqręmdęfceęu mpi ĦĘf ggąsrqęmu lęml ęs njga_rceąmbcĥęl beęk _i cqęns pj_qqęc_qgcp ęmk mbgwęgfcęąsrspcĦ

Next lesson

Nt cpjm_bg eef ceqs `qapgnręmcp_rmp

Back to table of contents

Previous lesson

Nt cpjm_bg eef ceąmk n_pggml mcp_rmpq

Leave a comment... Put C++ code between triple-backticks (markdown style):```Your C++ co

Name*

Email*

@t_r_ppqejbmk ęf rrnqIJhep_t_r_pΑmk h ę pcęaml l carcbęrnęwns pęnpmt gbcbęck _gię bbpcqqH

Mmrgdwęk cę `ms repcnjgcqIJę ę

POST COMMENT

152 COMMENTS

Newest

X

X