TASK 1:

#include <iostream>

using namespace std;

class Integer

{

//integer class

private: //private member

int val = 5;

public:

void input()

{

cout << "Enter value:";

cin >> val;

}

string toString(int value)

{

//public method to convert to string

int temp = 0, temp1;

string out;

temp1 = value;

while (temp1 != 0)

{

temp = temp1 % 10;

temp1 = temp1 / 10;

switch (temp)

{

case 0:

out = " ZERO" + out;

break;

case 1:

out = " One" + out;

break;

case 2:

out = " Two" + out;

break;

case 3:

out = " Three" + out;

break;

case 4:

out = " Four" + out;

break;

case 5:

out = " Five" + out;

break;

case 6:

out = " Six" + out;

break;

case 7:

out = " Seven" + out;

break;

case 8:

out = " Eight" + out;

break;

case 9:

out = " Nine" + out;

break;

}

}

return out;

}

string toOctalString(int value)

{

//method to convert to octal

string out;

while (value != 0)

{

int temp = 0;

temp = value % 8;

value = value / 8;

switch (temp)

{

case 0:

out = " Zero" + out;

break;

case 1:

out = " One" + out;

break;

case 2:

out = " Two" + out;

break;

case 3:

out = " Three" + out;

break;

case 4:

out = " Four" + out;

break;

case 5:

out = " Five" + out;

break;

case 6:

out = " Six" + out;

break;

case 7:

out = " Seven" + out;

break;

case 8:

out = " Eight" + out;

break;

case 9:

out = " Nine" + out;

break;

}

}

return out;

}

void operator++()

{

++val;

}

Integer operator++(int)

{

Integer temp; //++ opeartor overloading

temp.val = val++;

return temp;

}

Integer operator+(Integer& v)

{

Integer temp; //+ opeartor overloading

temp.val = val + v.val;

return temp;

}

Integer operator-(Integer& v)

{

//- opeartor overloading

Integer temp;

temp.val = val - v.val;

return temp;

}

void operator==(Integer v)

{

//== opeartor overloading

if (val == v.val)

{

cout << "the variables are equal." << endl;

}

else

cout << "the variables are not equal." << endl;

}

void operator+=(Integer v)

{

//+= opeartor overloading

val += v.val;

}

void operator=(Integer v)

{

//= opeartor overloading

val = v.val;

}

Integer& operator\*(Integer v)

{

//\*opeartor overloading

Integer temp;

temp.val = val \* v.val;

return temp;

}

void display()

{

//simple display function

cout << val << endl;

}

};

int main()

{

Integer a, b, c; //Integer class objects

int x;

cout << "enter number : " << endl;

cin >> x;

cout << a.toString(x) << endl;

cout << a.toOctalString(x) << endl;

cout << "Post fix increment Unary operator: ";

b = a++;

b.display();

cout << "Pre fix increment Unary operator: ";

++a;

a.display();

a.input();

b.input();

cout << " operator(-) :";

c = a - b;

c.display();

cout << " operator(+) :";

c = a + b;

c.display();

cout << " operator(==) :";

a == b;

cout << " operator(+=) :";

a += b;

a.display();

cout << " operator(=) :";

a = b;

a.display();

cout << " operator(\*):";

c = a \* b;

c.display();

return 0;

}

