M. Ali. Arslan

19F-0348

**Lab # 13**

**Problem # 1**

#include<iostream>

#include<string>

using namespace std;

class Employee

{

public:

string Name;

int Age;

float salary;

Employee()

{

Name = "Imran";

Age = 19;

salary = 200000;

}

friend istream& operator>>(istream& input, Employee& x)

{

cout << "Enter your name: ";

input >> x.Name;

cout << "Enter your age: ";

input >> x.Age;

cout << "Enter your salary: ";

input >> x.salary;

return input;

}

friend ostream& operator<<(ostream& output, Employee& e)

{

output << "Your Name is: " << e.Name << endl;

output << "Your age is: " << e.Age << endl;

output << "Your salary is: " << e.salary << endl;

return output;

}

int operator!()

{

return (this->salary / 2);

}

};

class person

{

public:

string Name;

int Age;

person()

{

Name = "Hashir";

Age = 20;

}

void operator = (const Employee& e)

{

person obj1;

obj1.Name = e.Name;

obj1.Age = e.Age;

}

};

int main()

{

person ze;

Employee obj;

cin >> obj;

cout << obj;

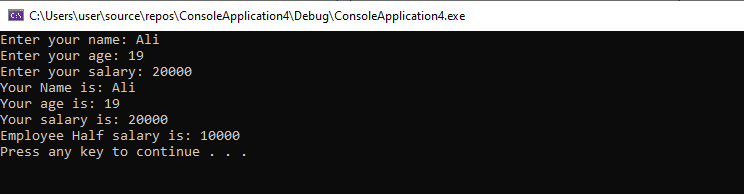
ze = obj;

cout << "Employee Half salary is: " << !obj << endl;

system("pause");

return 0;

}



**Problem # 2**

#include<iostream>

using namespace std;

class Employee {

string name;

int basic\_pay;

int medical\_allowance;

int house\_rent;

float total\_pay;

public:

Employee()

{

name = "";

basic\_pay = 0;

medical\_allowance = 0;

house\_rent = 0;

}

Employee(string a, int b, int c, int d)

{

name = a;

basic\_pay = b;

medical\_allowance = c;

house\_rent = d;

}

friend ostream& operator <<(ostream&, Employee&);

float operator!();

string operator+(Employee&);

bool operator==(Employee&);

bool operator<(Employee&);

int operator++(int);

int operator++();

int operator--(int);

int operator--();

};

ostream& operator<<(ostream& obj, Employee& c)

{

obj << "Name:" << c.name << endl;

obj << "Basic pay: " << c.basic\_pay << endl;

obj << "Medical Allowance: " << c.medical\_allowance << "%" << endl;

obj << "House rent: " << c.house\_rent << "%" << endl;

obj << "Tota Pay: " << !c << endl;

return obj;

}

float Employee::operator!()

{

total\_pay = basic\_pay + basic\_pay \* medical\_allowance / 100 + basic\_pay \* house\_rent / 100;

return total\_pay;

}

string Employee::operator+(Employee& c)

{

string a = name + c.name;

return a;

}

bool Employee::operator==(Employee& c)

{

return (total\_pay == c.total\_pay);

}

bool Employee::operator<(Employee& c)

{

return (total\_pay < c.total\_pay);

}

int Employee::operator++(int)

{

return basic\_pay++;

}

int Employee::operator++()

{

return ++basic\_pay;

}

int Employee::operator--(int)

{

return basic\_pay--;

}

int Employee::operator--()

{

return --basic\_pay;

}

int main()

{

Employee EMP1("Ali", 19000, 20, 10);

Employee EMP2("Arslan", 20000, 18, 15);

cout << "Employe 1 Data is: " << endl;

cout << EMP1;

cout << "Employe 2 Data is: " << endl;

cout << EMP2;

if (EMP1 == EMP2)

cout << "Salary Are Equal:" << endl;

else

cout << "Salary are not equal:" << endl;

if (EMP1 < EMP2)

cout << "Employee 1 is less than Emplayee 2" << endl;

cout << EMP1++ << endl;

cout << ++EMP1 << endl;

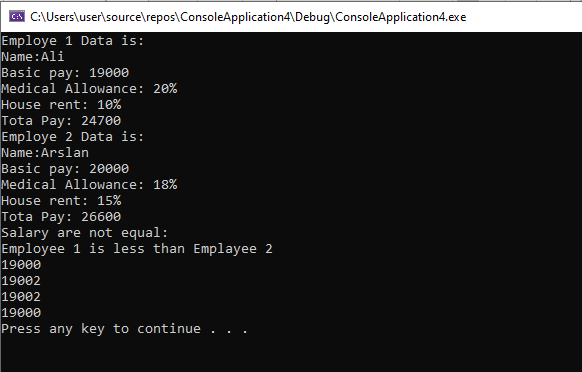
cout << EMP1-- << endl;

cout << --EMP1 << endl;

system("pause");

return 0;

}



**Problem # 3**

#include<iostream>

using namespace std;

class complex

{

public:

float a;

float b;

complex()

{

a = 0;

b = 0;

}

complex(float x, float y)

{

a = x;

b = y;

}

float operator +(complex&);

float operator \*(complex&);

friend float operator!(complex&);

complex operator--(int);

complex operator--();

complex operator++(int);

complex operator++();

friend ostream& operator<<(ostream& os, const complex& c);

};

float operator!(complex& c)

{

float mag;

mag = sqrt((((c.a) \* (c.a)) + ((c.b) \* (c.b))));

return mag;

}

float complex::operator+(complex& c)

{

float add;

add = (a + c.a) + (b + c.b);

return add;

}

float complex::operator\*(complex& c)

{

float mul;

mul = ((a \* c.a) - (b \* c.b)) + ((a \* c.b) + (c.a \* b));

return mul;

}

complex complex::operator++(int)

{

complex obj;

obj.a = a++;

obj.b = b++;

return obj;

}

complex complex::operator++()

{

complex obj1;

obj1.a = ++a;

obj1.b = ++b;

return obj1;

}

complex complex::operator--(int)

{

complex obj2;

obj2.a = a--;

obj2.b = b--;

return obj2;

}

complex complex::operator--()

{

complex obj3;

obj3.a = --a;

obj3.b = --b;

return obj3;

}

ostream& operator<<(ostream& os, const complex& c)

{

os << c.a << "+" << c.b << "i" << endl;

return os;

}

int main()

{

complex com1(7, 5);

complex com2(2, 13);

cout << "Magnitude of complex number 1 is: " << !com1 << endl;

cout << "Magnitude of complex number 2 is: " << !com2 << endl;

cout << "Sum of complex numbers is: " << endl;

cout << com1 + com2 << endl;

cout << "Multiplication of complex numbers" << endl;

cout << com1 \* com2;

cout << "Post increment of 1" << endl;

cout << com1++;

cout << "Pre increment of 1" << endl;

cout << ++com1;

cout << "Post decrement of 2" << endl;

cout << com2--;

cout << "Post increment of 2" << endl;

cout << --com2;

system("pause");

return 0;

}

