Muhammed Bedir ULUGAY 1901042697 [illegible]

CSE-241-501 Midterm Session III

"I hereby pledge that I will stricly adhere to academic integrity codes and the work done on this examination is salely my own and I will not receive/give any help from/to anybody or source during this examination"

Design and implement a C++ class to represent a Person

Your class will have the ----

- A constructor that takes all parameters (name, lastName, age, gender)
- A function return exsisting Person obj.
- Overload << Overload == !=

Overload pre and post increment operators that increment person age 1

- Overload < compares person age - Error checking

```
class Person {
  public:
      Person();
      Persion(string name, string lastName, int age, string gender);

      static int getCounter();
      void setAge(int x);
      friend ostream& operator<<(ostream& sout, const Person& obj);

      bool operator==(const Person &obj) const;

      bool operator!=(const Person &obj) const;

      Person operator++();

      Person operator++(int);

      bool operator<(const Person & obj) const;
  private:
      string name;
      string lastName;
      int age;
      string gender;
      static int counter;
};
```

```
int   Person :: counter = 0;

Person::Person() : name(""), lastName(""), age(-1), gender("")
 { counter ++; }
Person :: Person (string name, string lastName, int age, string gender)
        : name (name), lastName (lastName), age (age), gender (gender)
 {counter ++; }

 int  Person :: getCounter () {
        return  counter;
 }
 ostream&  operator<< (ostream & sout, const  Person& obj) {
      sout << obj.name <<  "   " << obj. lastName  <<  "\n";
      sout <<  obj.age <<  "\n";
      Sout <<  obj. gender << "\n";

      return  sout;
 }
bool  Person :: operator == (const  Person & obj ) const {
      return ( name ==  obj.name   &&  lastName == obj. lastName  &&
               age == obj.age     &&  gender == obj. gender );
}
bool  Person :: operator != (const Person &obj)  const{
      return  !(*this == obj);
}
Person   Person :: operator ++ () {
       age ++;
       return  *this;
}
Person   Person :: operator ++ (int  ignore) {
     Person  tmp (*this);
      ++(*this);
     return tmp;
```

```
bool Person::operator<(const Person& obj) const {
      return   age< obj.age;
}
void Person::setAge (int x) {  //Error checking
     if (x<0){
         cout<< "Age cannot be negative" << endl;
         return;
     }
     age = x;
}

int main () {

   Person a ("Ali ", "Gümüş, 23, "Male");
   Person b ("Veli", "Atın ,10, "Male");
   cout<<a;   cout <<b;
   if (a < b)
     cout "a is younger \n";
   else
     cout "a is not younger \n";

   if (a == b)
     cout "They are equal";

   else
     cout "not equal";

   cout<< Person::getStatic();

}
```