```
class Set: public List {
public:
    Set();                                          //Constructor
    Set(istream& i);                                //Constructor, scan stream i into the Set
    ~Set();                                         //Destructor
    void Intersection(Set& s1,Set& s2);             //s1 ∩ s2
    void Union(Set& s1,Set& s2);                    //s1 ∪ s2
    void Difference(Set& M,Set& S);                 //M − S = {m | m ∈ M and m ∉ S}
};
```

Figure 1. class *Set*.

1. **class** *Set* is derived from **class** *List*. **class** *Set* defines *Set* operations over the list given by **class** *List*.
2. Constructor *Set* explicitly calls the constructor for list to create a list on which the set is based. An empty *set* (*List*) is created.
3. Constructor *Set***(***istream***&** *i***)** creates a set having the unique integers in the file whose name is given by parameter *fn*.
4. Destructor ~*Set* implicitly calls destructor ~*List* and deletes all elements of the set (list).
5. Member function *Print* formats and prints the set according to the specifications given in project p06.
6. Member function *Intersection* forms the intersection of sets *s1* and *s2*.
7. Member function *Union* forms the union of sets *s1* and *s2*. Note that *Union* is capitalized to avoid the conflict with the reserve word **union**.
8. Member function *Difference* finds the difference of set *m(inuend)* – set *s(ubtrahend)*. All elements that are members of both *minuend* and *subtrahend* are removed from *minuend* to form the difference.

```
Set::Set():List(){}
```

Figure 2. Constructor *Set*.

```
//-------------------------------------------------------------------------------------------------------------------
//Member function Intersection finds the intersection of sets s1 and s2
//The intersection is composed of elements that are common to both sets.
//This implementation assumes that this set is empty prior to the invocation of member function
//Intersection.
//-------------------------------------------------------------------------------------------------------------------
void Set::Intersection(Set& s1,Set& s2)
{   Empty();
    for (s1.First();!s1.IsEol();s1.Next()) {
        int v=s1.ElementValue();
        if (s2.IsMember(v)) Insert(v);
    }
}
```

Figure 3. Member function *Intersection*.