問11　次のJavaプログラムの説明及びプログラムを読んで，設問1，2に答えよ。

（Javaプログラムで使用するAPIの説明は，この冊子の末尾を参照してください。）

〔プログラムの説明〕

名前及び住所からなる住所録のエントリの追加，削除及び検索を行う住所録管理プログラムであり，次のクラスからなる。

(1) クラス Name は，姓及び名をそれぞれ文字列で保持する。

(2) クラス Address は，郵便番号及び住所を保持する。郵便番号は，上位3桁と下位4桁をそれぞれ int 型の整数で保持する。住所は，文字列で保持する。

(3) クラス AddressBook は，住所録である。入れ子クラス AddressBook.Entry は，住所録のエントリであり，Name と Address のインスタンスを保持する。クラス AddressBook は，エントリを追加及び削除するメソッドをもつ。

クラス Name，Address，及び AddressBook.Entry のインスタンスは，検索可能である。検索処理を支援するために，次のインタフェースを定義する。

(1) インタフェース SearchCriteria は，検索条件を示すデータ型であり，メソッドをもたない。

(2) インタフェース Searchable は，これを実装するクラスが検索条件を与えて検索可能であることを示す。検索するときは，メソッド meets を呼び出す。

クラス Name 及び Address は，それぞれ入れ子クラス Criteria を定義し，その Criteria を検索条件とするインタフェース Searchable を実装する。

(1) 入れ子クラス Name.Criteria は，引数 familyName 及び givenName でそれぞれ姓及び名を検索条件として指定する。検索条件に含めないときは，null を指定する。

(2) クラス Name は，インタフェース Searchable を実装する。メソッド meets は，引数で与えられた criteria で実装されたメソッドを呼び出し，このクラスのインスタンスが検索条件に合致するかどうか調べる。

(3) 入れ子クラス Address.Criteria は，引数 postalCode3，postalCode4 及び addr でそれぞれ郵便番号の上位3桁，下位4桁及び住所を検索条件として指

定する。検索条件に含めないときは，postalCode3 及び postalCode4 は負の値，addr は null を指定する。住所については，部分文字列が一致する場合も合致しているとみなす。例えば，“京都”は“東京都”の部分文字列なので，合致するとみなす。

(4) クラス Address は，インタフェース Searchable を実装する。メソッド meets は，引数で与えられた criteria で実装されたメソッドを呼び出し，このクラスのインスタンスが検索条件に合致するかどうか調べる。

クラス AddressBook のメソッド meetsAnyOf は，引数で与えられたインタフェース SearchCriteria のどれかに合致するエントリの集合を返す。メソッド meetsAllOf は，引数で与えられた SearchCriteria の全てに合致するエントリの集合を返す。

入れ子クラス AddressBook.Entry は，インタフェース Searchable を実装する。メソッド meets は，引数で与えられた SearchCriteia の具体的な型によって，name 又は addr のメソッド meets を呼び出し，この AddressBook.Entry のインスタンスが引数で指定された検索条件に合致するかどうか調べる。

なお，クラス Name，Address，AddressBook.Entry は，インタフェース Set で使用できるように，クラス Object のメソッド equals 及び hashCode を上書きしているものとする。また，各コンストラクタ及びメソッドの引数は正しいものとする。

クラス Test は，この住所録プログラムのテストプログラムである。メソッド main を実行すると，次の結果が得られた。

```
[技術 太郎: 〒225-1234 横浜市青葉区, 試験 一朗: 〒980-9876 仙台市青葉区, 情報 太郎: 〒102-4567 東京都千代田区]
[技術 太郎: 〒225-1234 横浜市青葉区]
```

図 1　テストプログラムの実行結果

〔プログラム 1〕

```
public class Name implements Searchable<Name.Criteria> {
   private final String familyName, givenName;

   public static class Criteria implements [ a ] {
      private final String familyName, givenName;
```

```
        public Criteria(String familyName, String givenName) {
            this.familyName = familyName;
            this.givenName = givenName;
        }
        private boolean areMetBy(Name name) {
            return (familyName == null
                    || name.familyName.equals(familyName))
                && (givenName == null
                    || name.givenName.equals(givenName));
        }
    }

    public Name(String familyName, String givenName) {
        this.familyName = familyName;
        this.givenName = givenName;
    }

    public String getFamilyName() { return familyName; }

    public String getGivenName() { return givenName; }

    public boolean meets(Criteria criteria) {
        return criteria.areMetBy(this);
    }

    public String toString() {
        return familyName + " " + givenName;
    }
}
```

〔プログラム2〕

```
public class Address implements Searchable<Address.Criteria> {
    private final int postalCode3, postalCode4;
    private final String addr;

    public static class Criteria implements SearchCriteria {
        private final int postalCode3, postalCode4;
        private final String addr;

        public Criteria(int postalCode3, int postalCode4,
                        String addr) {
            this.postalCode3 = postalCode3;
            this.postalCode4 = postalCode4;
            this.addr = addr;
        }
        private boolean areMetBy(Address address) {
            return (postalCode3 < 0
                    || postalCode3 == address.postalCode3)
                && (postalCode4 < 0
                    || postalCode4 == address.postalCode4)
```

```
            && (addr == null
                || address.addr.contains(addr));
      }
   }

   public Address(int postalCode3, int postalCode4, String addr) {
      this.postalCode3 = postalCode3;
      this.postalCode4 = postalCode4;
      this.addr = addr;
   }

   public boolean meets(Criteria criteria) {
      return criteria.areMetBy(this);
   }

   public String toString() {
      return String.format("〒%03d-%04d %s",
                           postalCode3, postalCode4, addr);
   }
}
```

〔プログラム3〕

```
import java.util.HashSet;
import java.util.Set;

public class AddressBook {
   private Set<Entry> book = new HashSet<Entry>();

   public void add(Name name, Address addr) {
      book.add(new Entry(name, addr));
   }

   public void remove(Entry entry) { book.remove(entry); }

   public Set<Entry> meetsAnyOf(SearchCriteria... criteria) {
      Set<Entry> result = new HashSet<Entry>();
      for (SearchCriteria sc : criteria) {
         for (Entry entry : book) {
            if (entry.meets(sc))
               [    b    ];
         }
      }
      return result;
   }

   public Set<Entry> meetsAllOf(SearchCriteria... criteria) {
      Set<Entry> result = new HashSet<Entry>(book);
      for (Entry entry : book) {
         for (SearchCriteria sc : criteria) {
            if (!entry.meets(sc)) {
```

```
                c      ;
                break;
            }
        }
    }
    return result;
}

public static class Entry implements
                        Searchable<SearchCriteria> {
    private final Name name;
    private final Address addr;

    public Entry(Name name, Address addr) {
        this.name = name;
        this.addr = addr;
    }

    public Name getName() { return name; }

    public Address getAddress() { return addr; }

    public boolean meets(    d    criteria) {
        if (criteria instanceof Name.Criteria)
            return name.meets((Name.Criteria) criteria);
        if (criteria instanceof    e    )
            return addr.meets((    e    ) criteria);
        return false;
    }

    public String toString() { return name + ": " + addr; }
}
}
```

〔プログラム4〕

```
public interface SearchCriteria {
}
```

〔プログラム5〕

```
public interface Searchable<T extends SearchCriteria> {
    public boolean meets(T criteria);
}
```

〔プログラム6〕

```
public class Test {
    public static void main(String[] args) {
```

```
        AddressBook addrbook = new AddressBook();
        addrbook.add(new Name("情報", "太郎"),
                     new Address(102, 4567, "東京都千代田区"));
        addrbook.add(new Name("情報", "花子"),
                     new Address(102, 4567, "東京都千代田区"));
        addrbook.add(new Name("技術", "太郎"),
                     new Address(225, 1234, "横浜市青葉区"));
        addrbook.add(new Name("試験", "一朗"),
                     new Address(980, 9876, "仙台市青葉区"));
        System.out.println(addrbook.meetsAnyOf(
                new Name.Criteria(null, "太郎"),
                new Address.Criteria(-1, -1, "青葉区")));
        System.out.println(addrbook.meetsAllOf(
                new Name.Criteria(null, "太郎"),
                new Address.Criteria(-1, -1, "青葉区")));
    }
}
```

設問1　プログラム中の［　　］に入れる正しい答えを，解答群の中から選べ。

aに関する解答群

ア　SearchCriteria　　イ　SearchCriteria<Name>

ウ　SearchCriteria<String>　　エ　Searchable

オ　Searchable<Name>　　カ　Searchable<String>

b，cに関する解答群

ア　book.add(entry)　　イ　book.add(sc)　　ウ　book.remove(entry)

エ　book.remove(sc)　　オ　result.add(entry)　カ　result.add(sc)

キ　result.remove(entry)　ク　result.remove(sc)

d，eに関する解答群

ア　Address.Criteria　　イ　Criteria　　ウ　Entry.Criteria

エ　Name.Criteria　　オ　SearchCriteria

設問2　クラス Test において，住所録 addrbook に登録されている全エントリを取得する方法として正しい答えを，解答群の中から二つ選べ。ここで，プログラム中の［　　］には，正しい答えが入っているものとする。

解答群

ア　addrbook.meetsAllOf(new Name.Criteria("", ""))

イ　addrbook.meetsAllOf(new Name.Criteria(null, null))

ウ　addrbook.meetsAllOf(null)

エ　addrbook.meetsAnyOf(new Name.Criteria("", ""))

オ　addrbook.meetsAnyOf(new Name.Criteria(null, null))

カ　addrbook.meetsAnyOf(null)