アルゴリズムとプログラミング期末試験

問１． Java 言語とアルゴリズムに関する次の文章の空欄 (a) 〜 (q) を埋めよ。

- あるクラスのオブジェクトをコンストラクト (作成) する場合には，キーワード (a) を使って，そのクラスの (b) を呼び出す。
- あるクラスに属するメソッドを区別する (c) は，メソッドの名前と引数の (d) の並びからなる。また，同じ名前で引数の (d) の並びが異なるメソッドを作ることを，メソッドの (e) と呼ぶ。
- あるクラス A を元に，拡張・変更した新しいクラス B を宣言することを，クラス A を (f) して，クラス B を宣言すると言う。このとき，クラス A を (g) あるいは (h) と呼び，クラス B を (i) あるいは (j) と呼ぶ。クラスを (f) するときには，キーワード (k) を利用する。
- 親クラスと同じ (c) を持ったメソッドを子クラスで定義することを，メソッドの (l) と呼ぶ。
- 親クラスのコンストラクタを参照したいときなどには，キーワード (m) を使う。
- データを昇順あるいは降順に並び替えるソートの方法には，(n)，バブルソート，クイックソート，(o) などがある。バブルソートの場合，データ量が 10 倍になれば，計算量はおよそ (p) 倍になる。ソートにかかる時間は，一般的には，バブルソートを使う方がクイックソートを使うよりも (q) い。

問２． 次ページのプログラムは，成績を入力してその順番にデータを並び替えるプログラムである。

(1) クラス Gakusei は，１人分の学生のデータを格納するためのクラスである。

- フィールド point は，4 つの試験の成績を格納するための整数型配列で，フィールド maxPoint には 4 つの成績の最大値，フィールド sumPoint には 4 つの成績の和，フィールド name には学生名を格納する。
- Gakusei() はコンストラクタで，整数型配列のための長さ 4 の領域を確保し point に代入する。
- setName(String inName) は，名前 inName と同じ文字列を作成し name に代入するメソッドである。
- setPoint(int p0, int p1, int p2, int p3) は，４つの試験の成績 p0, p1, p2, p3 を，point の各要素に格納すると共に，それらの最大値と和を計算し，maxPoint，sumPoint に格納するメソッドである。
- ge(Gakusei gaku) は，自分の成績が gaku の成績より優れているか等しい場合に true を，そうでない場合は false を返り値とするメソッドである。ここで，成績が優れているとは，maxPoint が大きい場合，あるいは，両者の maxPoint が等しい場合は，sumPoint が大きいことである。また，成績が同じであるとは，maxPoint と sumPoint の両方が等しいことである。

(2) クラス Gakkyu は，学級の学生のデータを格納し，ソートするためのクラスである。

- フィールド gakusei は，Gakusei のオブジェクト参照型配列 (各要素が Gakusei のオブジェクトを参照している配列) である。
- Gakkyu(int nGakusei) はコンストラクタである。まず，Gakusei のオブジェクト参照型配列のための，長さ nGakusei の領域を確保し gakusei に代入する。次に，nGakusei 個の Gakusei のオブジェクトをコンストラクトし，gakusei の各要素に代入する。
- sortSelect() は，gakusei の内容を選択ソートによって，成績の降順 (配列で成績評価が高いものが先にくる順番) に並び替えるメソッドである (評価が同じ場合の順番は問わない)。
- setData(int gn, String inNm, int p0, int p1, int p2, int p3) は，gakusei[gn] に，学生の名前 inNm と，４つの試験の成績 p0, p1, p2, p3 のデータを格納するメソッドである。

(3) Kimatsu2 は，メインメソッドのためのクラスである。

このとき，次の問に答えよ。

(ア) 無駄なく，上の仕様を満たすために，プログラム中の下線部 (a)〜(h) に書き入れるべきものを答えよ。

(イ) このプログラムを実行したとき (コマンド java Kimatsu2 によって動かしたとき)，出力されるものを答えよ。
(確認：System.out.println() を呼んでいる箇所はプログラム中に一カ所だけである。)

```
class Gakusei {
  int[]  point;
  int    maxPoint, sumPoint;
  String name;
  Gakusei() {
    point = new int[4];
  }
  void setName(String inName) {
    name = new String(inName);
  }
  void setPoint(int p0, int p1, int p2, int p3) {
    point[0] = p0; point[1] = p1; point[2] = p2; point[3] = p3;
    maxPoint = sumPoint = point[0];
    for (int i = 1 ; i < 4 ; ++i) {
      if (maxPoint <  point[i]) maxPoint = point[i];
      sumPoint += point[i];
    }
  }
  boolean ge(Gakusei gaku) {
    if      (____________(a)____________) return true;
    else if (____________(b)____________) return false;
    else if (____________(c)____________) return true;
    else return false;
  }
}

class Gakkyu {
  Gakusei[]  gakusei;
  Gakkyu(int nGakusei) {
    gakusei = new Gakusei[nGakusei];
    for (int loop = 0 ; loop < nGakusei ; ++loop) ____________(d)____________
  }
  void sortSelect() {
    for (int i = 0 ; i < ____________(e)____________ ; ++i) {
      int     maxj = i;
      for (int j = i + 1 ; j < gakusei.length ; ++j) {
        if (____________(f)____________ ) ____________(g)____________
        }
      }
      System.out.println(gakusei[i].name + " のデータと " + gakusei[maxj].name+ "のデータを交換");
      ____________(h)____________
      gakusei[maxj] = gakusei[i];
      gakusei[i]    = tmp;
    }
  }
  void setData(int gn, String inNm, int p0, int p1, int p2, int p3) {
    gakusei[gn].setName(inNm); gakusei[gn].setPoint(p0, p1, p2, p3);
  }
}

class Kimatsu2 {
  public static void main(String[] args) {
    Gakkyu gakkyu = new Gakkyu(5);
    gakkyu.setData(0, "鈴宮",   96,  100, 100, 100); gakkyu.setData(1, "麻比奈", 80,  70,  60,  90);
    gakkyu.setData(2, "長戸",   100, 100, 100, 100); gakkyu.setData(3, "小泉",   90,  80,  80,  90);
    gakkyu.setData(4, "スミス", 60,   60,  60, 100);
    gakkyu.sortSelect();
  }
}
```

問 3. 右下に示すようなクラス図を，下記の Java プログラム Kimatsu03.java に実装することを考える．下記の設問に答えよ．

（1） プログラム中のコメント文を参照して，(ア) ～ (ウ) の下線部に入る適当な語句を記しなさい．
（2） クラスＺを（エ）に実装しなさい．ただし，フィールド変数 z(int 型)は，z = x * y で初期化され，アクセス修飾子は無指定とする．また，持っている全てのフィールドを表示するための，返り値を持たないメソッド printFieldValue()を実装せよ．
（3） コメント文を参照して適当な文を（オ）～（ク）に記せ．
（4） （ケ）はコンパイルエラーとなる．理由を述べ，正しく動作するように下線部を書き換えよ．
（5） プログラム(Kimatsu03.java)の実行結果を記せ．ただし（ケ）は正しく書き換えられているものとする．

Kimatsu03.java

```
class X {

  static int x = 0;

  X(){
    x++;
  }

  // printFieldValue()は、所属するクラスの全てのフィールド変数
  //の内容を表示するメソッド（表示フォーマットは自由）
  void printFieldValue(){
    System.out.println( ＿＿＿＿（ア）＿＿＿＿ );
  }

}
// クラス図を参照してクラス Y を実装せよ
class Y ＿＿＿＿（イ）＿＿＿＿ {

  int y = x * x;

  // printFieldValue()は、所属するクラスの全てのフィールド変数
  //の内容を表示するメソッド（表示フォーマットは自由）
  void printFieldValue(){
    System.out.println( ＿＿＿＿（ウ）＿＿＿＿ );
  }

}
// クラス図を参照してクラス Z を実装せよ。ただし、
// クラス Z のフィールド変数 z(int 型)は、z = x * y で初期化され、
// アクセス修飾子は無指定。また、所属するクラスの全てのフィールド変数
// の内容を表示するメソッド printFieldValue()を実装せよ。

 （エ）


class Kimatsu03 {
 public static void main(String[] args){

// クラス Z のインスタンスを生成し、
// クラス X のオブジェクト参照変数 o1 に代入せよ
＿＿＿＿（オ）＿＿＿＿;

// クラス Y のインスタンスを生成し、
// クラス X のオブジェクト参照変数 o2 に代入せよ
＿＿＿＿（カ）＿＿＿＿;

// クラス X のインスタンスを生成し、
// クラス X のオブジェクト参照変数 o3 に代入せよ
＿＿＿＿（キ）＿＿＿＿;

// o2 に対して printFieldValue()メソッドを呼べ
＿＿＿＿（ク）＿＿＿＿;

// o1 のフィールドを出力しようとして下記の文を書いたがコンパイルエラー
//となった。その理由を述べ、正しく動作するように下線部を書き換えよ。
(ケ)System.out.println( "o1.x=" + o1.x + "o1.y=" + o1.y );

 }
}
```

クラス図