AAAAAAAAAAAAAAAAAAAAAAAAAAAA

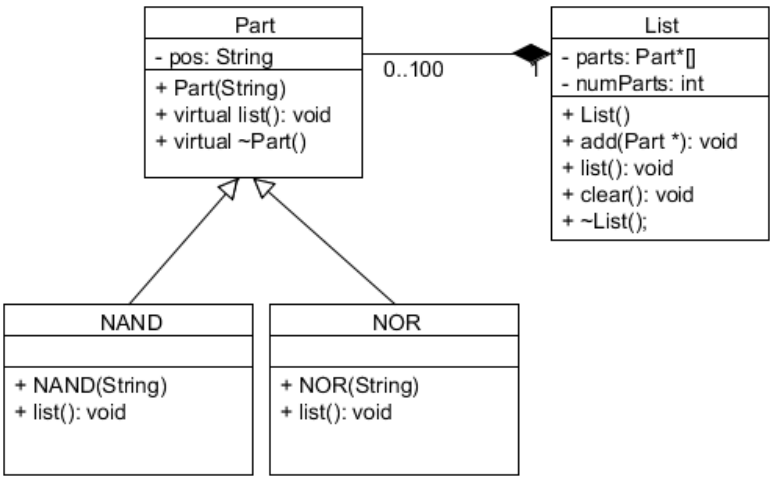
Programozás alapjai II. 3. ellenőrző dolgozat. 2012.04.05. Kurz/**Terem**: G2/	**20 perc**
Név: **Neptun:**	Összpont:

1. feladat 3 pont

Digitális áramkörök tervezését támogató rendszerben az **alkatrészlistát** úgy szeretnénk elkészíteni, hogy minden alkatrész mellé kirajzoljuk az áramkör jelét is. Minden alkatrésznek van egy szöveges pozíciójele is (pl: G12). Tudjuk, hogy az alkatrészek száma soha nem haladja meg a 100-at. Először csak *NAND* és *NOR* kapukat akarunk használni, de később újabb alkatrészekkel akarjuk bővíteni a rendszert. Az alkatrészek (*Part*) pozíciójelét *String*-ként kell tárolni. A rendszerben megvalósítandó műveleteket egy kódrészlettel mutatjuk be:

```
List parts;                    // Ez lesz az alkatrészlista
parts.add(new NAND("G12"));    // G12 jelű NAND kapu hozzáadása a listához
parts.add(new NOR("G45"));     // G45 jelű NOR kapu hozzáadása a listához
parts.list();                  // alkatrészlista kiíratása
parts.clear();                 // alkatrészlista törlése
```

Feltételezheti, hogy a *List* osztályból példányosított objektumot nem akarjuk paraméterként átadni és értékadás jobb, ill. bal oldalán sem szerepel. **Tervezzen** meg és **rajzoljon** fel egy olyan osztályhierarchiát, ami alkalmas az alkatrészek tárolására és könnyen bővíthető. Az osztálydiagramban jelölje az adattagok és metódusok láthatóságát is! A *String* osztályt nem kell lerajzolni, arra típusként hivatkozzon!



2. feladat 3 pont

Deklarálja a *List*, *Part*, és *NOR* osztályokat! **Csak** a *List* és a *NOR* osztályok konstruktorát, valamint a *List* osztály az *add()*, és *list()* metódusát **valósítsa** meg!

```
class Part {
      String pos;
public:
      Part(String);
      virtual void list();
      virtual ~Part();
};

class NOR : public Part {
public:
      NOR(String s)
            :Part(s) {}
      void list();
};
```

```
class List {
      Part* parts[100];
      int numParts;
public:
      List() :numParts(0) {}
      void add(Part* p) {
            parts[numParts++] = p;
      }
      void list() {
            for (int i = 0; i < numParts; i++)
                  parts[i]->list();
      }
      void clear();
      ~List();
};
```