Aufgabe 1

	Vorteile	Nachteile
Queue	Beliebig viele Objekte einfügen möglich	Lesen / Löschen „nur“ für 1. Element möglich, Einfügen „nur“ am Ende
Stack	Beliebig viele Objekte einfügen möglich	Lesen / Löschen „nur“ für das „oberste“ Element möglich, Einfügen ebenso
Liste	Beliebig viele Objekte einfügen möglich Lesen für beliebige Elemente möglich Einfügen / löschen an beliebiger Stelle	Für den Zugriff auf ein Elemente muss man jedesmal vom Anfang der Liste mit next weitergehen.
Array	Lesen für beliebige Elemente möglich Einfügen / löschen an beliebiger Stelle Schneller Zugriff über den Index	Anzahl der Elemente ist nicht änderbar. Beim Löschen entstehen Lücken. Zum Einfügen muss man evtl. Elemente verschieben

Hinweis:
Arrays sind „einfacher“ zu programmieren als dynamische Datenstrukturen und werden daher oft bevorzugt; das sollte jedoch kein Entscheidungskriterium sein.

Aufgabe 2

```
public class Adressbuch
{
   private List<Kontakt> aListe;

   public Adressbuch()
   {
      aListe = new List();
   }

   public void einfügen(Kontakt pk)
   {
      aListe.append(pk);
   }

   public Kontakt suchen(String pname)
   {
      Kontakt kAkt;
      aListe.toFirst();
      while (aListe.hasAccess() == true)
      {
         kAkt = aListe.getContent();
         if (pname.equals(kAkt.getNachname()))
         {
            return kAkt;
         }
         aListe.next();
      }
      return null;
   }
```

```
   public void namenAusgeben()
   {
      Kontakt kAkt;
      int anzahl;

      anzahl = 0;
      aListe.toFirst();
      while (aListe.hasAccess() == true)
      {
         kAkt = aListe.getContent();
         Console.println(kAkt.getNachname() + " " + kAkt.getVorname());
         aListe.next();
         anzahl++;
      }
      Console.println("(" + anzahl + " Kontakte im Adressbuch.)");
      Console.readln();
   }
}
```

Aufgabe 3

```
public class AdressVerwaltung
{
   private Adressbuch aBuch;

   public AdressVerwaltung()
   {
      aBuch = new Adressbuch();
   }

   public void kontaktAnlegen()
   {
      String n, v, t, e;
      Kontakt k;

      Console.println("== Neuen Kontakt hinzufügen ==");
      Console.print("Nachname: ");
      n = Console.readln();
      Console.print("Vorname:  ");
      v = Console.readln();
      Console.print("Telefon:  ");
      t = Console.readln();
      Console.print("Email:    ");
      e = Console.readln();

      k = new Kontakt(n, v, t, e);
      aBuch.einfügen(k);

      Console.println("Kontakt hinzugefügt.");
      Console.readln();
   }
```

```
   public void kontaktDetails()
   {
      String name;
      Kontakt k;

      Console.println("== Kontaktdetails ==");
      Console.print("Nachname: ");
      name = Console.readln();

      k = aBuch.suchen(name);
      if (k != null)
      {
         Console.println(k.getNachname() + ", " + k.getVorname());
         Console.println("Telefon: " + k.getTelefon());
         Console.println("Email:   " + k.getEmail());
      }
      else
      {
         Console.println("Es gibt keinen Kontakt mit diesem Namen.");
      }
      Console.readln();
   }

   public void main()
   {
      int wahl;
      do
      {
         Console.clear();
         Console.println("== Adressbuch - Hauptmenü ==");
         Console.println("Wählen Sie einen der folgenden Punkte:");
         Console.println("1. Neuer Kontakt");
         Console.println("2. Kontakte auflisten");
         Console.println("3. Kontaktdetails");
         Console.println("4. Programm beenden");
         wahl = Console.readInt();
         if (wahl == 1)
         {
            kontaktAnlegen();
         }
         else if (wahl == 2)
         {
            aBuch.namenAusgeben();
         }
         else if (wahl == 3)
         {
            kontaktDetails();
         }
         else if (wahl != 4)
         {
            Console.println("Falsche Eingabe");
         }
      } while (wahl != 4);

      Console.println("Auf Wiedersehen!");
   }
}
```

Aufgabe 4

```
public void einfügen(Kontakt pk)
{
   Kontakt kAkt;
   aListe.toFirst();
   while (aListe.hasAccess())
   {
      kAkt = aListe.getContent();
      if (pk.getNachname().compareTo(kAkt.getNachname()) < 0)
      {
         aListe.insert(pk);
         return;
      }
      aListe.next();
   }
   aListe.append(pk);
}

public Kontakt suchen(String pname)
{
   Kontakt kAkt;

   aListe.toFirst();
   while (aListe.hasAccess() == true)
   {
      kAkt = aListe.getContent();
      if (pname.equals(kAkt.getNachname()))
      {
         return kAkt;
      }
      if (pname.compareTo(kAkt.getNachname()) < 0)
      {
         return null;
      }
      aListe.next();
   }
   return null;
}
```

Aufgabe 5

Wenn die Liste des Adressbuchs für andere Klassen zur Verfügung stände, könnten diese Klassen mithilfe der Methoden der Klasse List an beliebiger Stelle Kontakte einfügen. Das würde dann die alphabetische Reihenfolge durcheinanderbringen. Eine Folge wäre, dass die Such-Funktion nicht mehr korrekt funktioniert, da diese abbricht, sobald der gesuchte Name alphabetisch VOR dem aktuellen Namen der Liste liegt. Aus diesem Grund sollte die Liste nur von Methoden der Klasse Adressbuch verändert werden.