Assignment 1: Classes and Objects

1. Consider the following 'nonsense class'.

```
class A
{ public A() { n = 0; }
  public A(int a) { n = a; }
  public void f() { n++; }
  public void g() { f(); n = 2 * n; f(); }
    public int h() { return n; }
  public void k() { System.out.println(n); }
  private int n;
}
```

Identify the constructors, mutator functions, and accessor functions. What kind of variable is `n`?

---

2. With the nonsense class from the preceding exercise, determine what the following program prints.

```
public static void main(String[] args) {
  A a = new A();
  A b = new A(2);
  A c = b;
  A d = new A(3);
  a.f();
  b.g();
  c.f();
  d.g();
  d.k();
  A e = new A(a.h()
            + b.h()
            + c.h()
        );

}
```

Work through the program by hand. Do not actually compile and run the program. Then run it and compare the results.

---

3. Implement all the methods of the following class:

```
class Person {
  public Person() {

….
  }
  public Person(String givenName, int yearOfBirth) {
    ……….
  }
  public String getName() {
        …..
  }
  public String changeName(String name) {
    …….. 
  }
  public int getAgeInYears(int currentYear) {
    ………
  }
  private String name;
  private int birthdayYear;
  public static void main(String[] args) {

    ……
  }
}
```

Write a small test program that creates and works with objects of class `Person` as well.

Design exercises:

4. Implement a class `Address`. An address has

- a house number,
- a street,
- an optional apartment number,
- a city,
- a state and a
- postal code.

Supply two constructors:

- one with an apartment number
- and one without.

Supply a `print` function that prints the address with the street on one line and the city, state, and postal code on the next line.

Supply a method `compareTo` that tests whether one address comes before another when the addresses are compared by postal code.

---

5. Implement a class `Account`. An account has

- a `balance`,
- functions to `add`
- and `withdraw` money,
- and a function to `inquire` the current balance.

Pass a value into a constructor to set an initial balance.

If no value is passed the initial balance should be set to `$0`.

Charge a `$5` penalty if an attempt is made to withdraw more money than available in the account.

Enhance the `Account` class to compute `interest` on the current balance.

---

6. Implement a class `Bank`. This bank has two objects

- `checking`
- and `savings`

of the type `Account` that was developed in the preceding exercise.

Implement four instance methods:

```
deposit(double amount, String account)

withdraw(double amount, String account)

transfer(double amount, String account)

printBalances()
```

Here the account string is `"S"` or `"C"`. For the deposit or withdrawal, it indicates which account is affected. For a transfer it indicates the account from which the money is taken; the money is automatically transferred to the other account.