# Unit 5 Progress Check: FRQ

**1.** SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.

Assume that the classes listed in the Java Quick Reference have been imported where appropriate.
Unless otherwise noted in the question, assume that parameters in method calls are not `null` and that methods are called only when their preconditions are satisfied.
In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods will not receive full credit.

The following class represents an invitation to an event. The variable `hostName` represents the name of the host of the event and the variable `address` represents the location of the event.

```
public class Invitation
{
     private String hostName;
     private String address;

     public Invitation(String n, String a)
     {
         hostName = n;
         address = a;
     }
}
```

(a) Write a method for the `Invitation` class that returns the name of the host.

Write the method below.

(b) Write a method for the `Invitation` class that accepts a parameter and uses it to update the address for the event.

Write the method below.

(c) Write a method for the `Invitation` class that will accept the name of a person who will be invited as a string parameter and return a string consisting of the name of the person being invited along with name of the host and location of the event.

For example, if the host name is `"Karen"`, the party location is `"1234 Walnut Street"`, and the person invited is `"Cheryl"`, the method should return a string in the following format.

```
Dear Cheryl, please attend my event at 1234 Walnut Street. See you
then, Karen.
```

Write the method below. Your implementation must conform to the example above.

(d) A student has written the following one-parameter constructor to be included in the `Invitation` class. The

## Unit 5 Progress Check: FRQ

method is intended to construct a new `Invitation` object that sets `address` to the value of the parameter and sets `hostName` to the default name `"Host"`. The constructor does not work as intended.

```
public Invitation(String address)
{
     address = address;
     hostName = "Host";
}
```

Write a correct implementation of the one-parameter `Invitation` constructor that avoids the error in the student's implementation.

Write the method below.

## Unit 5 Progress Check: FRQ

**2.** SHOW ALL YOUR WORK. REMEMBER THAT PROGRAM SEGMENTS ARE TO BE WRITTEN IN JAVA.

Assume that the classes listed in the Java Quick Reference have been imported where appropriate.
Unless otherwise noted in the question, assume that parameters in method calls are not null and that methods are called only when their preconditions are satisfied.
In writing solutions for each question, you may use any of the accessible methods that are listed in classes defined in that question. Writing significant amounts of code that can be replaced by a call to one of these methods will not receive full credit.

This question involves the implementation of the `PasswordGenerator` class, which generates strings containing initial passwords for online accounts. The `PasswordGenerator` class supports the following functions.

Creating a password consisting of a specified prefix, a period, and a randomly generated numeric portion of specified length
Creating a password consisting of the default prefix `"A"`, a period, and a randomly generated numeric portion of specified length
Reporting how many passwords have been generated

The following table contains a sample code execution sequence and the corresponding results.

## Unit 5 Progress Check: FRQ

| Statements | Possible Value Returned (blank if no value returned) | Comment |
|---|---|---|
| `PasswordGenerator pw1 = new PasswordGenerator(4, "chs");` | | Passwords generated by the `pw1` object are composed of the prefix `"chs"`, a period, and 4 randomly-generated digits. |
| `pw1.pwCount();` | `0` | No passwords have been generated yet. |
| `pw1.pwGen();` | `"chs.3900"` | A possible password generated by the `pw1` object |
| `pw1.pwGen();` | `"chs.1132"` | A possible password generated by the `pw2` object |
| `pw1.pwCount();` | `2` | Two passwords have been generated. Both contain the prefix `"chs"` and 4 digits. |
| `PasswordGenerator pw2 = new PasswordGenerator(6);` | | Passwords generated by the `pw2` object are composed of the default prefix `"A"`, a period, and 6 randomly generated digits. |
| `pw2.pwCount();` | `2` | Two passwords have been generated. Both contain the prefix `"chs"` and 4 digits. |
| `pw2.pwGen();` | `"A.843055"` | A possible password generated by the `pw2` object |
| `pw2.pwCount();` | `3` | Three passwords have been generated. Two contain the prefix `"chs"` and 4 digits, and the third contains the default prefix `"A"` and 6 digits. |
| `pw1.pwCount();` | `3` | Three passwords have been generated. The same value is returned by `pwCount` for all objects of the `PasswordGenerator` class. |

Write the complete `PasswordGenerator` class. Your implementation must meet all specifications and conform to the example.