AP Computer Science
Dr. Paul Bailey

Project 8 - Hangman Game
September 20, 2019

Due Wednesday, October the 2nd, 2019, at or before 2359 military time. Zip and send source code of all Java programs to via email to `paul.bailey@basised.com`. Please rename the zip file so it has your name on it. Also, put your name in a comment on the top of each program.

The goal of this project is to construct a Hangman game. This will serve as an introduction to arrays; in this case, arrays of characters and arrays of strings.

Program 1. Create a new JDK directory and call it `Hangman`. Inside this directory, create a new Java program `Hangman1.java`. Type in, compile, and run the following.

```
public class Hangman1
{
    private static String[] words =
    {
        "AARDVARK",
        "BEETLE",
        "CAMEL",
        "DINGO",
        "ELEPHANT",
        "FLAMINGO",
        "GOOSE",
        "HOG",
        "IGUANA",
        "JAGUAR"
    };

    public static void main(String[] args)
    {
        Prog1();
    }

    public static void Prog1()
    {
        int k = 0;
        while (k < words.length)
        {
            String s = words[k];
            if (s.length() < 1) break;
            System.out.printf("%3d) %s\n", k+1, s);
            k++;
        }
    }
}
```

Expect the output

```
  1) AARDVARK
  2) BEETLE
  3) CAMEL
  4) DINGO
  5) ELEPHANT
  6) FLAMINGO
  7) GOOSE
  8) HOG
  9) IGUANA
 10) JAGUAR
```

Program 2. Copy `Hangman1.java` into `Hangman2.java`, and change the class name accordingly.

(a) Type an import statement at the top of the program for a random number generator:

```
import java.util.Random;
```

(b) Declare a class variable for a random number generator:

```
private static Random random = new Random(0);
```

The parameter 0 is know as the "seed", and causes the randomizer to produce the same sequence of random numbers upon each execution.

(c) Create a method to return a random word:

```
public static String randomWord()
{
    // use global random object to return a random word
}
```

(d) Modify the main method to call `Prog2a`:

```
public static void Prog2a()
{
    for (int i = 1; i <= 10; i++)
    {
        String s = randomWord();
        System.out.printf("%3d) %s\n", i, s);
    }
}
```

Expect the output

```
 1) AARDVARK
 2) IGUANA
 3) JAGUAR
 4) HOG
 5) FLAMINGO
 6) DINGO
 7) BEETLE
 8) BEETLE
 9) JAGUAR
10) ELEPHANT
```

(e) Create a static method `public static String blankWord(String s)` which returns a string consisting of `s.length` underscores:

```
public static String blankWord(String s)
{
    char[] tA = new char[s.length()];
    // Loop through the array of characters and
    // set each slot equal to an underscore character
    return new String(tA);
}
```

(f) Test your method with `Prog2b`:

```
public static void Prog2b()
{
    for (int i = 1; i <= 10; i++)
    {
        String a = randomWord();
        String b = blankWord(a);
        System.out.printf("%3d) %-20s %-20s\n", i, a, b);
    }
}
```

Expect the output

```
 1) AARDVARK             ________
 2) IGUANA               ______
 3) JAGUAR               ______
 4) HOG                  ___
 5) FLAMINGO             ________
 6) DINGO                _____
 7) BEETLE               ______
 8) BEETLE               ______
 9) JAGUAR               ______
10) ELEPHANT             ________
```

(g) Create a method `public static String adjustWord(String s, String t, char g)` which tests if s and t have the same length, and if they do, replaces the i^{th} letter of t with g whenever the i^{th} letter of s equals g:

```
public static String adjustWord(String s, String t, char g)
{
    if (s.length() == t.length())
    {
        char sA[] = s.toCharArray();
        char tA[] = t.toCharArray();
        // Loop through sA, if sA[i] == g, set tA[i] = g
        t = new String(tA);
    }
    return t;
}
```

(h) Test the new method with this `Prog2c`:

```
public static void Prog2c()
{
    for (int i = 1; i <= 10; i++)
    {
        String a = randomWord();
        String b = blankWord(a);
        b = adjustWord(a, b, 'A');
        System.out.printf("%3d) %-20s %-20s\n", i, a, b);
    }
}
```

Expect the output

```
 1) AARDVARK             AA___A__
 2) IGUANA               ___A_A
 3) JAGUAR               _A__A_
 4) HOG                  ___
 5) FLAMINGO             __A_____
 6) DINGO                _____
 7) BEETLE               ______
 8) BEETLE               ______
 9) JAGUAR               _A__A_
10) ELEPHANT             _____A__
```

Program 3. Copy `Hangman2.java` into `Hangman3.java`.

(a) Type an import statement at the top of the program for a scanner:

```
import java.util.Scanner;
```

(b) Declare a class variable for a scanner:

```
private static Scanner scanner = new Scanner(System.in);
```

(c) Create a function `public static char guess()`:

```
public static char guess()
{
    String s;
    char c;
    while (true)
    {
        System.out.print("Guess: ");
        s = scanner.nextLine().toUpperCase();
        if (s.length() == 1)
        {
            c = s.charAt(0);
            if ((c >= 'A' && c <= 'Z') || c == '.') break;
        }
    }
    return c;
}
```

(d) Test the program with this `Prog3a`:

```
public static void Prog3a()
{
    char g = '@';
    while (g != '.')
    {
        g = guess();
        System.out.println("> " + g);
    }
}
```

(e) Create a method `public static void play(String a)` which begins to play the game Hangman with the supplied string a. This version of `play` should:

(1) Create a string variable b.

(2) Set b to a blank string of the length of a.

(3) In a loop, display b and ask for a letter g.

(4) Appropriately adjust b with that letter.

(5) When a and b contain the same word, print "You win!" and quit the loop.

(f) Test the program with this `Prog3b`:

```
public static void Prog3b()
{
    String a = randomWord();
    play(a);
}
```

(g) Insert a constant class variable at the top of the class: `private static final int maxWrong = 6`.

(h) Modify the `play` method:

(1) Create a string variable w; this will be a list of the incorrectly guessed letters.

(2) Test if the guess g is in a but not in b.

(3) If the guess is in a but not in b, adjust b.

(4) If the guess is not in a and not in w, append the letter to w in order to keep track of the incorrectly guessed letters.

(5) Show w along with b in the same print statement.

(6) If the maximum number of incorrect guesses is reached, print "You LOOSE!" and return.

(i) Test the program with `Prog3b`.

Program 4. Copy `hangman3.java` into `hangman4.java`. Modify the program to use character based "graphics" which show the man actually hanging. Start with the gallows. The first wrong guess displays the head, the next the body, then left and right arms, left and right legs.

```
--------
|      |
|      O
|     /|\
|     / \
|
|
=========
```