```java

// Create a program to calculate and output your final grade in PROG1400.
//
// The program should use primitive variable types only (e.g. no arrays, etc.).
// The values of each assignment should be stored as constants and
// can be set to any reasonable value (since you don't have those grades yet).
//
// You can ignore the optional assignment at the end of the course.

public class Assignment1a {

    public static void main(String[] args) {

        //declaring variables
        int firstNum = 0;
        int secondNum = 1;
        int maxNumbers = 20;
        int outputNum;
        int i;

        //printing the first number
        System.out.print(secondNum + " ");

        //using a loop to print out the numbers
        for (i = 2; i <= maxNumbers; ++i) {
            outputNum = firstNum + secondNum;
            System.out.print(outputNum + " ");
            firstNum = secondNum;
            secondNum = outputNum;

        }


    }

}
```

```
 1
 2 //  Fibonacci numbers are a series of numbers where the first two numbers are 1,
 3 //  and each subsequent number is the sum of the two previous numbers.
 4 //
 5 //  e.g.  1 1 2 3 5 8 13 ….
 6 //
 7 //  Create a program that calculates and outputs the first 20 numbers in the
 8 //  sequence on a single line.
 9
10 public class Assignment1b {
11
12     public static void main(String[] args) {
13
14         // making an array of characters to contain the modified alphabet
15         char[] oldAlphabet = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M',
16                               'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z'};
17         char[] newAlphabet = {'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z',
18                               'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M'};
19
20         //storing a single word as a constant variable
21         String str = "secret";
22
23         char encryptedText = ' ';
24
25         // looping through each letter in the string
26         int count = 0;
27         System.out.println("Decrypted Text: ");
28         System.out.println(str);
29         System.out.println("Encrypted Text: ");
30         for (int i=0; i<str.length(); i++){
31             // making each character a single character and holding it into "ch"
32             char ch = str.charAt(i);
33
34             // checking if the character is equal to the character input
35             if (ch >= 'a' && ch <= 'm') {
36                 count++;
```

```
37                  encryptedText = (char) (ch + 13);
38              }
39              else if (ch >= 'n' && ch <= 'z') {
40                  count++;
41                  encryptedText = (char) (ch - 13);
42              }
43              System.out.print(encryptedText);
44          }
45
46
47
48
49      }
50  }
```

```java

//  ROT13 cipher is a very simple encryption cipher that rotates the alphabet by
//  13 letters to change “A” to “N”, “B” to “O”, “C” to “P”, “X” to “K”,
//  “Y” to “L”, “Z” to “M”, etc.
//
//  e.g. “secret” would be encrypted to “frperg”
//
//  Create a program that takes a single word in lowercase letters, stored as a
//  constant variable, and outputs it as a ROT13 encrypted version.

import java.util.Scanner;

public class Assignment1c {

    public static void main(String[] args) {
        Assignment1c RotProgram = new Assignment1c();

        Scanner s = new Scanner(System.in);

        //asking the user for an input
        System.out.print("Please put a sentence in: ");
        String str = s.nextLine();

        String changedString = RotProgram.changeString(str);

        //displaying the unencrypted and encrypted strings
        System.out.println("Unencrypted: " + str);
        System.out.println("Encrypted: " + changedString);

    }

    //String encryption
    private String changeString(String str) {
        String output = "";
        for (int i=0; i<str.length(); i++) {
            char changedChar = newChar(str.charAt(i));
            output += changedChar;
        }
        return output;
    }

```

```
42      public char newChar(char inputChar) {
43          int numAlphabet = 26;
44          char[] alphabet = {'a', 'b', 'c', 'd', 'e', 'f', 'g
    ', 'h', 'i', 'j', 'k', 'l', 'm',
45                             'n', 'o', 'p', 'q', 'r', 's', 't
    ', 'u', 'v', 'w', 'x', 'y', 'z'};
46
47          char changedChar = inputChar;
48          //Checking if the current letter is uppercase or
    lowercase
49          boolean uppercase = !(changedChar == (Character.
    toLowerCase(changedChar)));
50
51          for (int i=0; i<numAlphabet; i++) {
52              //converting the inputChar into lowercase
53              if (Character.toLowerCase(inputChar) ==
    alphabet[i]) {
54                  int newSpotInAlphabet = i + 13;
55
56                  if (newSpotInAlphabet >= numAlphabet) {
57                      newSpotInAlphabet -= numAlphabet;
58                  }
59
60                  //Changing the originally uppercase
    characters back to uppercase
61                  if (uppercase) {
62                      changedChar = Character.toUpperCase(
    alphabet[newSpotInAlphabet]);
63                  }else
64                      changedChar = alphabet[
    newSpotInAlphabet];
65              }
66          }
67          return changedChar;
68
69      }
70  }
```

```
 1
 2 //  Based on the previous question, expand your program to
   encrypt a word,
 3 //  phrase or sentence, which may contain spaces,
   punctuation or different case
 4 //  letters.
 5 //
 6 //  e.g.
 7 //  “This is a story about a man named Jed.” is changed to
 8 //  “Guvf vf n fgbel nobhg n zna anzrq Wrq.”
 9 //
10 //  Note that the letter case, the spacing and the
   punctuation match the
11 //  original text.
12
13 import java.util.ArrayList;
14 import java.util.Scanner;
15
16 public class Assignment1d {
17
18     public static void main(String[] args) {
19
20         Scanner s = new Scanner(System.in);
21
22         //Asking the user for the table name
23         System.out.print("Table Name: ");
24         String table = s.nextLine();
25
26         //declaring a variable if the user hits ENTER (
   empty string)
27         String empty = new String();
28         //storing the field values into an array
29         ArrayList<String> fieldValues = new ArrayList<
   String>();
30
31         //Asking the user for the field information
32         //Repeating the prompt until the user leaves it
   blank and hits ENTER
33         while (true) {
34             System.out.print("Field info (ENTER to end): "
   );
35             String field = s.nextLine();
36             //if the user hits ENTER (empty string), break
   the loop
37             if (field.equals(empty)) {
38                 break;
```

```
39              //otherwise, continue the loop
40              } else {
41                  fieldValues.add(field);
42                  continue;
43
44              }
45          }
46
47          //displaying the table name
48          System.out.println("CREATE TABLE " + table + " (");
49          //displaying the field information
50          //displaying each value in the fieldValues array
51          int i = 0;
52          for (i = 0; i < fieldValues.size(); i++)
53          {
54              System.out.println(fieldValues.get(i));
55          }
56          System.out.println(");");
57
58      }
59
60  }
```

```java

// Create a program that will generate a CREATE TABLE statement based on inputs
// from the user.
//
// The program should ask for the name of the table and the details of each
// field. The program should allow any number of fields and will output the
// complete CREATE TABLE statement once the user has completed entering in the
// information.

import java.util.Scanner;
import java.util.function.DoubleToIntFunction;

public class Assignment1e {

    public static void main(String[] args) {

        //declaring variables
        Scanner s = new Scanner(System.in);
        String [] months = {"Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"};
        int [] numMarriages = {70,93,99,151,224,324,737,925,729,483,128,175};
        int roundedTo = 100; //number is rounded off to the nearest hundreds
        int determiningNum = 50; //variable to determine how the number will be rounded off

        for (int i=0; i<numMarriages.length; i++) {
            //rounding each number to the nearest hundreds
            int remainder = numMarriages[i] % roundedTo;
            //determine if the remainder is greater than determining number
            if (remainder >= determiningNum) {
                numMarriages[i] += (roundedTo - remainder);
            } else {
                numMarriages[i] -= remainder;
            }
            //displaying the specific month
            System.out.print(months[i] + ": ");
            //determining the number of asterisks
            //each hundred is one asterisk
            int numOfAsterisks = numMarriages[i] /
```

```
37 roundedTo;
38              //displaying the asterisks
39              for (int i2=0; i2<numOfAsterisks; i2++){
40                  System.out.print("*");
41              }
42              System.out.println();
43          }
44
45      }
46
47 }
48
```