```java

/*
 * Mitch Feigenbaum
 * Period 5
 * February 1, 2022
 * On my honor, I pledge that I have neither given nor received unauthorized assistance on this assignment or test.
 */
import java.util.Calendar;
import java.util.Scanner;

/**
 * The FeigenbaumU4 class is a class that provides methods for checking if a
 * given month and date are valid, getting a calendar date in ordinal form,
 * getting an astrological sign with a message, and a CLI in which a client
 * can run the program (the main method).
 */
public class FeigenbaumU4 {
	/**
	 * The main method is a CLI that allows a client to interactively
	 * input their birthdate to get an output of their astrological sign
	 * with a unique message. The method first creates a Calendar object.
	 * This calendar object is used to store the current month and
	 * date into the todaymonth and todayday variables. A scanner is
	 * opened to collect the clients input. Then the program enters
	 * a do while loop that prompts the client to enter their birthdate,
	 * checks that the date is valid with the checkDate method, and stores
	 * The client's birthmonth and birthdate into separate integer
	 * variables. After the client's birthmonth and birthday are defined,
	 * the program checks if it is the client's birthday and if so
	 * initializes a birthday message. Finally a print format statement is
	 * created that displays the client's age in ordinal form (using the
	 * birthdate method), a birthday message in the event that the current
	 * date is their birthday, their astrological sign, and a unique
	 * message (using the sign method)
	 * <p>
	 *
	 * @param args added for semantics
	 * @see checkDate
	 * @see birthdate
	 * @see sign
	 */
	public static void main(String[] args) {
		Calendar today = Calendar.getInstance();
		int todaymonth = today.get(Calendar.MONTH) + 1;
		int todayday = today.get(Calendar.DAY_OF_MONTH);
		Scanner scanNum = new Scanner(System.in);
		int birthMonth;
		int birthDay;
		do {
			System.out.print("What month were you born in? (number): ");
			birthMonth = scanNum.nextInt();
			System.out.print("What day (number): ");
			birthDay = scanNum.nextInt();
			if (!checkDate(birthMonth, birthDay))
				System.err.println("Error: date does not exist.");
		} while (!checkDate(birthMonth, birthDay));
		String birthDayMessage = "";
		if (birthMonth == todaymonth && birthDay == todayday)
			birthDayMessage = "Happy Birthday to you!";
		System.out.printf("Your birthday is:\t%s\t%s%n%s",
				birthdate(birthMonth, birthDay), birthDayMessage, sign(birthMonth, birthDay));

	}

	/**
	 * The birthdate method is used to return a textual version of a date
	 * based on an integer representation of the month and date. First an
	 * array of all 12 calendar months is initialized. then a String
	 * format statement is returned that contains a textual representation
	 * of the month, and an ordinal representation of the date. (using the
	 * toOrdinal method)
	 * <p>
	 *
	 * @param m An integer representation of the month
	 * @param d An integer representation of the date
	 * @return a formatted string containing the month and date
	 * @see toOrdinal
	 */
	public static String birthdate(int m, int d) {
```

```
80                  String[] months = {
81                                  "January",
82                                  "February",
83                                  "March",
84                                  "April",
85                                  "May",
86                                  "June",
87                                  "July",
88                                  "August",
89                                  "September",
90                                  "October",
91                                  "November",
92                                  "December",
93                  };
94
95                  return String.format("%s %s", months[m - 1], toOrdinal(d));
96          }
97
98          /**
99           * The method sign provides an astrological sign and horoscope based
100          * on a provided month and date. The method iterates through a list of
101          * if statements that return the astrological sign and horoscope of a
102          * certain character (eg Aquarius) provided that the month and date
103          * are within a certain range of dates in which the character is
104          * valid. If the criteria of an if statement is fulfilled the method
105          * returns a string with an astrological sign and horoscope formatted
106          * with tabs for alignment. The tabstops are meant to be used in
107          * conjunction with the main method's print format statement which
108          * also displays the users birthday and a happy birthday message.
109          * <p>
110          *
111          * @param m An integer representation of the month
112          * @param d An integer representation of the date
113          * @return A string with the astrological sign and horoscope for the
114          *         provided month and day
115          * @see main
116          */
117         public static String sign(int m, int d) {
118                 if ((m == 1 && d >= 20) || (m == 2 && d <= 18))
119                         return "Your sign is:\t\tAquarius\n"
120                                 + "Horoscope:\t\tYou must hold back your desires and temptations.";
121                 if ((m == 2 && d >= 19) || (m == 3 && d <= 20))
122                         return "Your sign is:\t\tPisces\n"
123                                 + "Horoscope:\t\tSoon you will make amends with all you have wronged.";
124                 if ((m == 3 && d >= 21) || (m == 4 && d <= 19))
125                         return "Your sign is:\t\tAries\n"
126                                 + "Horoscope:\t\tYour home planet will be destroyed.";
127                 if ((m == 4 && d >= 20) || (m == 5 && d <= 20))
128                         return "Your sign is:\t\tTaurus\n"
129                                 + "Horoscope:\t\tHave faith in the plan.";
130                 if ((m == 5 && d >= 21) || (m == 6 && d <= 20))
131                         return "Your sign is:\t\tGemini\n"
132                                 + "Horoscope:\t\tThe best days of your life are already over.";
133                 if ((m == 6 && d >= 21) || (m == 7 && d <= 22))
134                         return "Your sign is:\t\tCancer\n"
135                                 + "Horoscope:\t\tYou will hold back your own success.";
136                 if ((m == 7 && d >= 23) || (m == 8 && d <= 22))
137                         return "Your sign is:\t\tLeo\n"
138                                 + "Horoscope:\t\tYou will achieve great success in the textile industry.";
139                 if ((m == 8 && d >= 23) || (m == 9 && d <= 22))
140                         return "Your sign is:\t\tVirgo\n"
141                                 + "Horoscope:\t\tThere is light at the end of your sorrow.";
142                 if ((m == 9 && d >= 23) || (m == 10 && d <= 22))
143                         return "Your sign is:\t\tLibra\n"
144                                 + "Horoscope:\t\tTerrible things are going to happen.";
145                 if ((m == 10 && d >= 23) || (m == 11 && d <= 21))
146                         return "Your sign is:\t\tScorpio\n"
147                                 + "Horoscope:\t\tYour days are numbered. Keep watch of all enemies.";
148                 if ((m == 11 && d >= 22) || (m == 12 && d <= 21))
149                         return "Your sign is:\t\tSagittarius\n"
150                                 + "Horoscope:\t\tSomeday you will travel into the multiverse.";
151                 if ((m == 12 && d >= 22) || (m == 1 && d <= 19))
152                         return "Your sign is:\t\tCapricorn\n"
153                                 + "Horoscope:\t\tYou will achieve inner peace.";
154                 return null;
155         }
156
157         /**
158          * The toOrdinal method can convert a number (from 1 to 39) from
159          * integer form into ordinal form. Based on a number's position in a
160          * certain range (eg 11-19), an element is chosen from an array which
```

```
161          * represents a number's ordinal form. If a number is not composed of
162          * only one word (eg twenty-sixth), a format string is returned
163          * containing the numbers tenth place, a dash, and the singular
164          * number that represents the digit in its ones place.
165          * <p>
166          *
167          * @param d a date 1-39
168          * @return an ordinal number
169          */
170         public static String toOrdinal(int d) {
171                 if (d > 10 && d < 20) {
172                         String[] teens = {
173                                         "eleventh",
174                                         "twelfth",
175                                         "thirteenth",
176                                         "fourteenth",
177                                         "fifteenth",
178                                         "sixteenth",
179                                         "seventeenth",
180                                         "eighteenth",
181                                         "nineteenth",
182                         };
183                         return teens[d - 11];
184                 }
185                 if (d % 10 == 0) {
186                         String[] tenths = {
187                                         "tenth",
188                                         "twentieth",
189                                         "thirtieth",
190                         };
191                         return tenths[d / 11];
192                 }
193                 String[] singleDays = {
194                                 "first",
195                                 "second",
196                                 "third",
197                                 "fourth",
198                                 "fifth",
199                                 "sixth",
200                                 "seventh",
201                                 "eighth",
202                                 "ninth",
203                 };
204                 if (d < 10)
205                         return singleDays[d - 1];
206                 String[] tens = {
207                                 "twenty",
208                                 "thirty",
209                 };
210                 return String.format("%s-%s", tens[d / 10 - 2], singleDays[d % 10 - 1]);
211         }
212
213         /**
214          * The checkDate method verifies whether a current month and date are
215          * possible calendar positions. It iterates through a series of
216          * impossible criteria that will return false if they are applicable
217          * to the provided month and day. The date is first checked that
218          * the month is between 1 and 12 and that the date is at least one.
219          * Then the date is checked to verify that it is not too large for the
220          * current calendar month. If none of the unverifiable criteria are
221          * met, the method will return true
222          * <p>
223          *
224          * @param m the current month in integer form
225          * @param d the current day in integer form
226          * @return true if the provided month and day are valid
227          */
228         public static boolean checkDate(int m, int d) {
229                 if (m < 1 || m > 12 || d < 1)
230                         return false;
231                 if ((m == 1
232                                 || m == 3
233                                 || m == 5
234                                 || m == 7
235                                 || m == 8
236                                 || m == 10
237                                 || m == 12)
238                                 && d > 31)
239                         return false;
240                 if ((m == 4
241                                 || m == 6
```

```
242                                 || m == 9
243                                 || m == 11)
244                                 && d > 30)
245                         return false;
246                 if (m == 2 && d > 29)
247                         return false;
248                 return true;
249         }
250 }
```