STUDENT DATABASE

Objectives: Storing Data in Arrays, Searching Data

Task: Write a program that will recognize invalid inputs when the user requests information about students in a class.

What will the application do?

- **2 Points:** Create 3 arrays and fill them with student information—one with **name**, one with **hometown**, one with **favorite food**
- **1 Point:** Prompt the user to ask about a particular student **by number**. Convert the input to an integer. Use the integer as the index for the arrays. Print the student's name.
- **1 Point:** Ask the user which category to display: **Hometown** or **Favorite food.** Then display the relevant information.
- **1 Point:** Ask the user if they would like to learn about another student.

Build Specifications:

- **1 Point: Validate user number:** Use an if statement to check if the number is out of range, i.e. either less than 1 or greater than the length of the arrays. If so, display a friendly message and let the user try again.
- **1 Point: Validate category:** Ask the user what information category to display: "Hometown" or "Favorite Food". Use an if statement to check that they entered either category name correctly. If they entered an incorrect category, display a friendly message and re-ask the question. (See hints below!)
- **1 Point: Array Length**: Use the first array's Length property in your code instead of hardcoding it.

Hints:

- Make sure the arrays are the same size.
- Let the user enter a number from 1 up to and including the length of the array. To get the index, subtract 1 from the number they entered.
- For the valid category: You might create a separate program to test out some code that uses a while loop and asks for either "Hometown" or "Favorite Food." And only finishes the loop if either of these two is entered. Once you have it working, copy the code over to your "real" code.
- Make it easy for the user – tell them what information is available
- Try to use good grammar. Make your messages polite.

Extra Challenges:

- **1 Point:** Provide an option where the user can see a list of all students.
- **2 Points:** Allow the user to search by student name (Good challenge but difficult!)
- **1 Point:** Category names: Allow uppercase and lowercase; allow portion of word such as "Food" instead of "Favorite Food"

Console Preview:

```
Welcome! Which student would you like to learn more about? Enter a number 1-9: 2

Student 2 is Jimmy Scrambles. What would you like to know? Enter "hometown" or
"favorite food":
age

That category does not exist. Please try again. Enter "hometown" or "favorite food":
hometown

Jimmy Scrambles is from Grand Rapids

Would you like to learn about another student? Enter "y" or "n": n
```