Lab 5 - If and Switch

Directions

- The labs are marked based on attendance and effort.
- It is your responsibility to ensure the TA records your progress by the end of the lab.
- Do each step of the lab in order.
- While completing these labs, you are encouraged to help your classmates and receive as much help as you like. Assignments, however, are individual work.
  You may not work on assignments during your lab time.
- If you complete the lab early, you should experiment with C++; however, you may leave if you prefer.
- If you do not finish the lab exercises during your lab time, you are encouraged to complete them later to finish learning the material. You will still receive full marks if you arrived on-time and put in your best effort to complete the lab.

1. Fun with if

- Create a program named `ifFun.cpp` which reads in an integer from the user. A set of sample outputs are given below to show how the program should operate.
- If the integer is even do both of the following:
  - print `"Even"` to the screen and
  - based on the integer's value, perform the **ONE** matching action in the following table:

User input value	Output to screen
< 0	`"Negative number."`
= 0	`"It's zero."`
> 0 but < 100	`"Small number."`
>= 100	`"Large number."`

- If the integer is odd, do **EACH** of the following:
  - Output `"Odd"`
  - If it is between 0 and 100 (inclusive), output `"Greater than or equal to 0, but less than or equal to 100."`
  - If it is divisible by both 3 and 5, output `"Divisible by 3 and 5."`
  - If it is not divisible by 7, output `"Not divisible by 7."`

Output on following page...

Note for output: When you run your program, it will only ask you for a single value. This output shows running the program multiple times just to show each of the possibilities.

Output 1: Sample output from multiple executions of the ifFun.cpp program.

```
Enter an integer: -20
Even
Negative number.

Enter an integer: 0
Even
It's Zero.

Enter an integer: 58
Even
Small number.

Enter an integer: 10008
Even
Large number.

Enter an integer: 5
Odd
Greater than or equal to 0, but less than or equal to 100.
Not divisible by 7.

Enter an integer: -45
Odd
Divisible by 3 and 5.
Not divisible by 7.

Enter an integer: 707
Odd

Enter an integer: 123456789
Odd
Not divisible by 7.

Enter an integer: -111
Odd
Not divisible by 7.
```

- **Understanding:**
  - You should be able to use an `if` statements with:
    - complex conditions (and / or);
    - use mod (%) to test for divisibility;
    - nested if-statements; and
    - if, else-if, else statements.

2. Find the bug

- It's tempting to believe a program works if:
  - It runs without error, and
  - It's output seems reasonable.
- However, to show a program works, you must methodically test it:
  - Test using inputs which should generate outputs which you can predict.
  - Validate the program against these predictions.
    Don't just think: *"The program's output isn't absurd, therefore the program's right."*
- Copy the following code into a program called `mealCost.cpp`. It has a bug. Can you find it?
  - Start your testing with a few inputs which are easy to check the answers.
    Hint: *0 is my favourite number, 1 is my second favourite number. Why?*

```
// This program determines total buffet luncheon cost when
// the number of guests and the per person cost are known.
// It contains a logic error.
// Program 4-29, by Gaddis.
#include <iostream>
#include <iomanip>
using namespace std;

const int ADULT_MEAL_COST = 6.25; // Child meal cost = 75% of this

int main()
{
     int    numAdults,         // Number of guests ages 12 and older
            numChildren;       // Number of guests ages 2-11
     double adultMealTotal,    // Cost for all adult meals
            childMealTotal,    // Cost for all child meals
            totalMealCost;

     // Get number of adults and children attending
     cout << "This program calculates total cost "
          << "for a buffet luncheon.\n";
     cout << "Enter the number of adult guests (age 12 and over): ";
     cin  >> numAdults;
     cout << "Enter the number of child guests (age 2-11): ";
     cin  >> numChildren;

     // Calculate meal costs
     adultMealTotal = numAdults * ADULT_MEAL_COST;
     childMealTotal = numChildren * ADULT_MEAL_COST * .75;
     totalMealCost  = adultMealTotal + childMealTotal;

     // Display total meal cost
     cout << fixed << showpoint << setprecision(2);
     cout << "\nTotal buffet cost is $" << totalMealCost << endl;
     return 0;
}
```

- **Understanding:** How to carefully test a program is correct.

3. Switch

Write a program called `foodMenu.cpp` which:

- Displays a menu (see output sample below), and asks the user to select an option 1-4.
- Use a `switch` statement, based on the user's selection, to calculate a sub-total.
  - Combo 1 is $3.50
  - Combo 2 is $15.72
  - Combo 3 is $1.10
  - Combo 4 the user types in a full line of text describing their order (use `getline()`).
    - Then, charge the customer $0.50 per letter in their input.
      (Yes, the cost of the meal is proportional to how long the description is; the cook reads really slowly!)
    - You can get the length of a string `someString` using:
      ```
      int strLength = someString.size();
      ```
    - Note: If you are declaring a variable inside a `switch` statement's case, you'll need to encase it in a block; without the block you will get an error. For example, use:
      ```
      case 1: {
       int i = 10;
       cout << i;
      }
      ```
    - Use `cin.ignore()` as required to manage the extra `'\n'` with `getline()`.
  - Add a `default` case to catch user errors by displaying an error message.
- Add a 10% tip to the order. (Hey! The automated console is giving the customer great service! The programer should get a tip, right?)
- Sample output on next page...

- Output 1:

```
Welcome to Self-serve Diner:
============================
Which combo would you like to order?
1. Triple-decker burger and small fry.
2. Supersized XXXL salad and diet jolt.
3. Chocolate cake and small fry.
4. Custom order.
: 2

Subtotal: 15.72
Tip: 1.57
Total: 17.29
```

- Output 2:

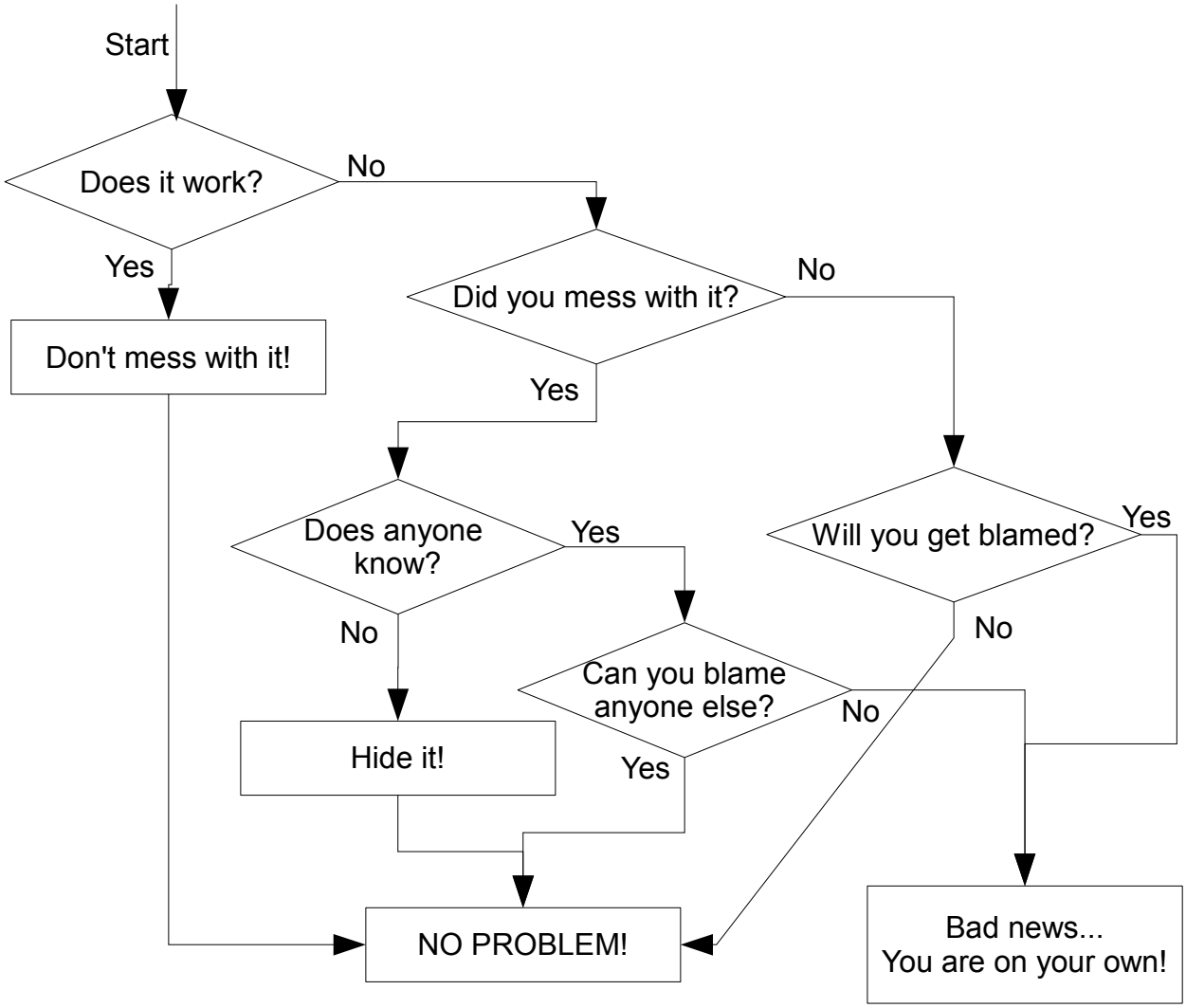
```
Welcome to Self-serve Diner:
============================
Which combo would you like to order?
1. Triple-decker burger and small fry.
2. Supersized XXXL salad and diet jolt.
3. Chocolate cake and small fry.
4. Custom order.
: 4
Enter your custom order ($0.50 per character!)
: Cold pizza and warm coke.

Subtotal: 12.50
Tip: 1.25
Total: 13.75
```

- **Understanding:**
  - Use the `switch` statement.
  - Use the `getline()` statement with `cin.ignore()`.

4. Problem Solving

Write a program which helps the user solve problems. Your program should guide them through the following flow chart using nested if statements.

- Diamonds represent a question you should ask the user. Responses should be "yes" or "no". You may assume the user typed either "yes" or "no" correctly (no error checking required).
- Rectangles represent statements you should output to the user.



- Hint:
  - Create a Boolean variable to track if there is a problem or not. At the end of your program, check this value to choose between "NO PROBLEM" or "Bad news..."
- Sample Output:

```
Does it work? [yes/no]: no
Did you mess with it? [yes/no]: yes
Does anyone know? [yes/no]: no
Hide it!
NO PROBLEM!
```

5. Skills and Understanding

You should now be able to answer all the "understanding" questions in the previous sections. Complete the following to get credit for the lab:

- Show the TA the following:
  - Your operational programs which complete all of the above tasks.
  - The TA may ask you to explain any section of the lab, or answer any of the "Understanding" questions.
- **Nothing** is to be submitted electronically or in hard-copy for this lab.

