

Problem Set 2

Due before midnight on Tuesday, January 29, 2008.

1 Assignment Goals

1. Learn how to do numeric input and output to the terminal in C.
2. Learn how to validate input.
3. Learn how to use the numeric types `int` and `double`.
4. Learn how to define constants.
5. Learn how to use functions from the math library.
6. Learn how to use the basic control structures `if`, `break`, and the unterminated loop `for(;;)`.
7. Think through the development process going from requirements to working code.
8. Develop and carry out a test plan for your program.

2 Bakery

Wedding Bells Bakery is in the business of baking wedding cakes. It sells two styles of cakes, round and square. Each style can be made in various sizes in order to accommodate the anticipated number of guests.

Customers generally have no idea how big a cake is needed or which shape would better accommodate their needs. You are to write a program to help Wedding Bells Bakery assist their customers in choosing a cake. In particular, the customer will say how many guests are expected and what size portions they want to serve. The program will find the smallest standard round and the smallest standard square cake that accommodate the customer's needs. For each cake, it will print out the size, number of servings, and cost.

2.1 WBB Standard Cake Sizes

WB cakes are made from stacks of successive layers of a given shape. Each layer is 1.5 inch thick. The smallest round layer is 6 inches in diameter. The smallest square layer is 6 inches on a side. Each successive layer is 2 inches larger than the previous. Thus, a 12 inch round cake would have 4 layers of sizes 6, 8, 10, and 12, respectively. Square cakes are similar, except that each layer is square. Round layers and square layers are never mixed in the same cake.

2.2 Servings

Each serving is cut from a single layer in the shape of a right prism whose base is a simple closed curve. Since all layers are the same thickness, the volume of a serving (in cubic inches) is proportional to the area of the enclosed surface (in square inches), so we talk about serving sizes in terms of the surface area of the piece rather than its volume.

The area of a round layer of diameter d is $\pi d^2/4$, and the area of a square layer of side length s is s^2 . There are many ways to obtain the value of π . One simple one is to use the fact that $\cos(\pi/2) = 0$; hence $\pi = 2 \cos^{-1}(0)$, where \cos^{-1} is the arccosine function.

If the desired serving size is s and the surface area of a layer is A , we assume that layer can be cut into $k = \lfloor A/s \rfloor$ pieces, each of size s (although the shapes may vary).¹ Any remaining partial serving is discarded.

The number of servings in a multilayer cake is just the sum of the number of servings that can be obtained from each layer.

2.3 Pricing

The cost of a layer is \$2 plus \$0.50 per square inch of cake for a round cake, and \$2 plus \$0.55 per square inch for a square cake (higher because square cakes are more difficult to remove from the cake pans cleanly). The cost of a cake is the total cost of its layers. The final cost of the cake is rounded to the nearest dollar.

3 Program Details

Your program should read two numbers from the user: the number of guests (a non-negative integer) and the desired serving size (a real number in the interval $[1.0, 8.0]$). The program should check each input for validity and prompt the user to re-enter the value in case of error.

The number of guests should be stored in a variable of type `int`. The desired serving size should be stored in a variable of type `double`.

For each type of cake (round and square), the program should find the smallest sized cake that can serve all of the guests, and it should calculate how many servings that cake will yield and what it will cost. Finally, the program should print the following output:

- The expected number of guests;
- The desired serving size;
- The size, yield, and cost of the smallest adequate round cake;
- The size, yield, and cost of the smallest adequate square cake.

Your program will need some functions from the C math library: `acos()`, `floor()` and `round()`. To use them, you must include `math.h` in your source file, and you must give the switch `-lm` (that's ℓ , not the digit 1) to the linker.

A sample transcript of a run of the program is shown in Figure 1.

¹For a real number x , the *floor* of x , written $\lfloor x \rfloor$, is the greatest integer less than or equal to x .

```
Welcome to the Wedding Bells Bakery cake-sizing program
How many guests do you expect? 123
How large a portion size do you want (4 square inches recommended)? 4

Here are your options for serving 123 guests with 4 square inch portions:
    A round cake of size 16 will yield 154 portions.  It costs $325.00.
    A square cake of size 14 will yield 135 portions.  It costs $307.00.

Thank you for choosing Wedding Bells Bakery.  Goodbye!
```

Figure 1: Sample run of the program.

4 Testing

You should construct a sequence of test cases that give some confidence in the correctness of your program. In particular, your test cases should:

- Test each of the input validation conditions: number of guests too small or too large, portion size too small or too large, or non-numeric input supplied.
- Cause round cakes and square cakes in each of the sizes 6, 8, and 10 to be output.
- Cause a situation where the best round cake and best square cake are different sizes.
- Cause a situation in which the best round cake is cheaper than the best square cake, and vice versa.

5 Submission

You should prepare a simple `Makefile` for compiling and linking your program with the required compiler switches as given in PS1. (Note that the `-I` switch is not needed for this assignment, but the others are.) To ease grading, your source file should be named `cake.c` and the executable named `cake`.

You should submit the source code files, `Makefile`, and a text file that gives your tests and the output of your program on each test. *All submitted files should have your name, the name of this course (CPSC 223b), the term (Spring 2008), and the assignment number (Problem Set 2) included in a comment at the top.*