

Due 2:00PM, Thursday, February 19th, 2009

---

Name \_\_\_\_\_

CSE Login \_\_\_\_\_

**Instructions** Follow instructions *carefully*, failure to do so may result in points being deducted. For questions 1 – 4: type your answers; for programming code, use a monotype font (like **Courier New**). Print your answers hardcopy and staple this cover page to the front of your assignment. Be sure to write your name on the front.

For each of the programs, place your source code into files named `homework02program01.c`, `homework02program02.c` and `homework02program03.c` respectively and turn them in using the webhandin available on the course webpage. You do not need to print out your source code for the programs.

You may discuss problems with your classmates, but all work must be your own. The CSE academic dishonesty policy is in effect (see [http://www.cse.unl.edu/undergrads/academic\\_integrity.php](http://www.cse.unl.edu/undergrads/academic_integrity.php)).

Question	Points	Score
1	10	
2	10	
3	10	
4	5	
5	20	
6	20	
7	25	
Total:	100	

1. 10 points Consider the following variables:

```
1 int a = 10;
2 int b = -5;
3 int c = 15;
4 int flag = 1;
```

Evaluate the following expressions given the values above. Indicate which parts of the expression (if any) are not evaluated due to *short-circuiting*.

- (a) `c == b - a || !flag`  
(b) `a != -a || !flag && c <= (-b * a)`  
(c) `!(a \% b) && b <= -5`  
(d) `(a > 15) || c < a + b`  
(e) `!!a == 10 && c >= 15 || b`
2. 10 points Convert each of the following from English statements to proper C statements.

- (a) `x` is less than `y` and `y` is greater than `z`  
(b) `x` is greater than `y` and `x` is less than or equal to `z`  
(c) `x` is true and `y` is not equal to `z` or `z` is not equal to 42  
(d) `x` is equal to `y` or `x` is equal to 1  
(e) `x` is less than or equal to `y` and `y` is less than or equal to `z`

3. 10 points Rewrite the following mathematical expressions using math functions from the standard C math library (assume all variables are of type `double`):

- (a)  $\sqrt{|z^3 - 2.0|}$   
(b)  $\frac{x^{\sin 2y}}{xy}$   
(c)  $a^2 + b^2 - 2ab \cos C$   
(d)  $e^x \sin y$   
(e)  $(\sin x)^2 + (\cos y)^3 - \sqrt{\frac{\tan x}{2}}$

4. 5 points Write a function called `distance` that takes as input four `double` values `x_1`, `y_1`, `x_2`, `y_2` that specify two Cartesian points  $(x_1, y_1)$  and  $(x_2, y_2)$  and returns the distance between them computed by using the formula:

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

5. 20 points **Program 1**

Write a program that will calculate and print bills for the city power company. The rates vary depending on whether the use is residential, commercial, or industrial. A code of **R** means residential use, a code of **C** means commercial use, and a code of **I** means industrial use. Any other code should be treated as an error. The rates are computed as follows:

R: \$7.80 plus \$0.1134 per kilowatt hour (kwh) used

C: \$67.50 for the first 1000 kwh and \$0.0765 for each additional kwh

I: \$76.50 for the first 1000 kwh, \$213.50 for the next 1500 kwh, and \$0.0545 for each additional kwh

Your program will prompt the user to enter an use code (**R,C,I** of type **char**) and the number of kwh used (of type **double**). Your program will simply display the amount due.

6. 20 points **Program 2**

The table below shows the normal boiling points of several substances. Write a program that prompts the user for the observed boiling point of a substance in degrees Celsius and identifies the substance if the observed boiling point is within 5% of the expected boiling point. If the data input is more than 5% higher or lower than any of the boiling points in the table, it should output **Unknown substance**.

<b>Substance</b>	<b>Boiling Point (C)</b>
Methane	-161.7
Butane	-0.5
Water	100
Nonane	150.8
Mercury	357
Copper	1187
Silver	2193
Gold	2660

Table 1: Expected Boiling Points

7. 25 points **Program 3**

Write a Program to create a table containing a customized loan amortization table. our program will prompt the user to enter the amount borrowed (the *principal*), the annual interest rate (format: 5.05 is 5.05%), and the number of monthly payments (*n*). To calculate the monthly payment, use the formula given in Programming Project 1 in Chapter 3 of your text. This payment must be rounded to the nearest cent. After the payment has been rounded to the nearest cent, the program will create a table that looks something like the following.

```
1 Principal: $1000.00
2 Annual Interest Rate: 9.00%
3 Term: 6 months
4 Monthly Payment: $171.07
5 Payment      Interest      Principal      Balance
6 1             $7.50          $163.57       $836.43
7 2             $6.27          $164.80       $671.63
8 3             $5.04          $166.03       $505.60
9 4             $3.79          $167.28       $338.32
10 5            $2.54          $168.53       $169.79
11 6            $1.27          $169.79       $0.00
12 Final Payment: $171.06
```

Each month, part of the payment is applied to the interest and the rest is applied to the principal. Because the payment and each month's interest are rounded, the final payment will be a bit different and must be calculated as the sum of the final interest payment and the final principal balance.