Project 1, Program Design

1. A public parking garage offers hourly parking with some flexibility and options at the following rates:

First 30 minutes is free;

Each additional 20 minutes $1;

And $12 daily max (24 hours).

Write a program *parking.c* that calculates and prints the charges for parking at the parking garage.

1) The user enters the number of total hours and minutes; the program prints the charge.
2) If the input is invalid, print a message and exit the program. Invalid inputs include the following
    a. number of hours or minutes is negative
    b. number of minutes is greater than 60
3) A floating point division of two integers can be obtained by a type casting:
    (double)integer1/integer2
4) ceil function in the math library might be useful. To use ceil function, compile with -lm:
    gcc -lm -Wall parking.c

Example input/output:

```
Enter hours parked: 5
Enter minutes parked: 46
Amount due ($): $12
```

Example input/output:

```
Enter hours parked: 0
Enter minutes parked: 34
Amount due ($): $1
```

Example input/output:

```
Enter hours parked: 47
Enter minutes parked: 28
Amount due ($): $24
```

Example input/output:

```
Enter hours parked: 50
```

```
Enter minutes parked: 2
Amount due ($): $29
```

2. The Chinese zodiac is based on the lunar calendar that assigns an animal and its reputed attributes to each year in a repeating 12-year cycle. The zodiac traditionally begins with the sign of the Rat, and followed by Ox, Tiger, Rabbit, Dragon, Snake, Horse, Sheep, Monkey, Rooster, Dog and Pig. Each sign has its own strengths and weaknesses, its own specific traits, desires and attitude towards life and people. For example, the personality traits for the monkey sign are entertaining, intelligent, optimistic, sociable, fickle, secretive, unpredictable.

To calculate the animal sign, divide the year of birth by 12 and obtain the remainder. Each remainder corresponds to an animal sign.

0: Monkey	1: Rooster	2: Dog	3: Pig
4: Rat	5: Ox	6: Tiger	7: Rabbit
8: Dragon	9: Snake	10: Horse	11: Sheep

In this program, you will calculate the animal sign for a year of birth entered by the user.

Example input/output:

Enter the year of birth: 2005

Output: The Chinese animal sign for 2005 is Rooster

1) Name your program chinese_zodiac.c
2) If the input is less than or equal to zero, output an error message and abort the program.
3) Use a switch statement for the determining the animal sign.

Before you submit:

1. Compile with –Wall. –Wall shows the warnings by the compiler. Be sure it compiles on ***student cluster (sc.rc.usf.edu)*** with no errors and no warnings.

gcc –Wall parking.c

gcc –Wall Chinese_zodiac.c

2. Be sure your Unix source file is read & write protected. Change Unix file permission on Unix:

   chmod 600 parking.c

   chmod 600 chinese_zodiac.c

3. Test your program with the shell script on Unix:

   chmod +x try_parking

   ./try_parking

   chmod +x try_zodiac

   ./try_zodiac

4. Download the program *parking.c and Chinese_zodiac.c* from student cluster and submit on Canvas>Assignments.

Grading

Total points: 100 (50 points each problem)

1. A program that does not compile will result in a zero.
2. Runtime error and compilation warning 5%
3. Commenting and style 15%
4. Functionality 80%

Programming Style Guidelines

The major purpose of programming style guidelines is to make programs easy to read and understand. Good programming style helps make it possible for a person knowledgeable in the application area to quickly read a program and understand how it works.

1. Your program should begin with a comment that briefly summarizes what it does. This comment should also include your **name**.
2. In most cases, a function should have a brief comment above its definition describing what it does. Other than that, comments should be written only *needed* in order for a reader to understand what is happening.
3. **Variable names** and function names should be sufficiently descriptive that a knowledgeable reader can easily understand what the variable means and what the function does. If this is not possible, comments should be added to make the meaning clear.
4. Use consistent **indentation** to emphasize block structure.
5. Full line comments inside function bodies should conform to the indentation of the code where they appear.
6. Macro definitions (#define) should be used for defining symbolic names for numeric constants. For example: `#define PI 3.141592`
7. Use names of moderate length for variables. Most names should be between 2 and 12 letters long.
8. Use either underscores or capitalization for compound names for variable: `tot_vol`, `total_volumn`, or `totalVolumn`.