C Programming lab
Batch2 Midterm Exam

Prepared by: M. Rao

Revision Date: September 29, 2014

Preliminaries

This exam has a total of 20 points. All tasks are all or nothing. There is no partial credit given for individual tasks. The duration of this exam is two hours. You must put all your programs in the correct directories and submit from the specified directory or you will receive zero credit.

Your are allowed to write the code in unindented form. Also no need of comments in your files. However make sure that your output should be in the format specified. For example, spaces, newlines and decimals should be followed as specified in the test cases. Your code will be checked for only output. Given a test case, your code should work as expected otherwise you will lose points for that task.

Create a directory named *batch2* that hangs off your `/home/IMT2013XXX` directory using 'mkdir' linux command. Create four directories inside the *batch2* directory: *separate*, *donation*, *bus*, and *complex* directories. You should see the following output:

```
4   ./separate
4   ./complex
4   ./bus
4   ./donation
4   .
20  .
```

The names must be as shown, but the order and the numbers do not matter.

Task 1: Separate vowels and consonants from a list of characters(5 points)

Develop a program in *separate.c* in *separate* directory which will take command line arguments consisting of all characters. The data parsed from the command line arguments are separated by a space. Your program should identify whether the character is a consonant or a vowel.and put these characters into *consonant.dat*, and *vowel.dat* files.

If your program is tested as follows

```
./separate a b c d e i o u
```

then your program should have the following data in vowel.dat file

```
a e i o u
```

and following data in consonant.dat file

```
    b c d
```

Remember to put a space between the characters in output files. Note that your program will be tested with different characters and different count of characters.

[HINT: If you need to write to a file, you can use fprintf statement with a 'w' mode]

Task 2: Displaying complex numbers(5 points)

Write a single function *compute* in *util.c* file in *complex* directory to evaluate real and imaginary values of a complex number which is given by $e^{ikz+\phi} = (e^{\phi}.cos(kz) + i.e^{\phi}.sine(kz))$. The values of ϕ, k and z will be passed as command line arguments in the following format.

./complex $< \phi > < k > < z >$

The *compute* function will take atmost four arguments as formal parameters. You need to call compute function from other file *complex.c*, where you will write *main* function. Link and compile both files to produce a *complex* executable. Your program should demonstrate the following output.

```
    ./complex 1 1 1
    1.47 + i2.29
```

Another example:

```
    ./complex 0 0 0
    1.00
```

Note that in the previous example, imaginary part comes out to be zero and hence only real value is displayed.

Note that you have to print the values only in main function in *complex.c* file. No printf's/fprintf's should be placed in *util.c* file. *complex.c* file will only call the function and display the result. No computation will is expected in *complex.c* file. Computation to find real and imaginary number needs to be done in *util.c* file in the *compute* function.

Again note that you have to write a single function in *util.c* file. If more than one function is written, your solution will not be considered for grading. Also if more than 4 formal parameters are used in function definition, then your solution will not be considered correct.

[HINT: You may use exp, cos, and sin built-in functions from existing math library. If you use these functions, remember to link math library while compiling]

Task 3: Bus (5 points)

Develop a program in *bus.c* file in *bus* directory. Your program should take two input files (*cost.dat* and *journey.dat*) from the command line argument and should calculate the total amount to be carried before travelling in the local bus services. Your program will be executed in the given below format.

```
./bus cost.dat journey.dat
```

The cost of each destination is mentioned with respect to a central point and is mentioned in the *cost.dat* file. Cost of the journey from the starting point to reach destination point is the difference between the cost of two points as mentioned in *cost.dat* file. The journey including starting and destination point is mentioned in *journey.dat* file. The *cost.dat* file has data in the following format as shown below:

```
<destination-name> <cost>
```

One such *cost.dat* file is shown below:

```
corporation 50.00
lalbagh 75.50
btm 80.50
electronic-city 110.00
```

The *journey.dat* file has data in the following format as shown below:

```
<starting-point> <destination-point>
```

One such *journey.dat* file is shown below:

```
lalbagh electronic-city
```

Your program is supposed to calculate the cost of reaching destination point from starting point. The total cost of your travelling for the above test case is

```
34.50
```

Another example with different list of *journey.dat* and *cost.dat*. One such *cost.dat* file is shown below:

```
corporation 10.00
kormangala  20.00
madiwala    30.00
hsr         40.00
ecospace    60.00
whitefield  70.00
```

One such *journey.dat* file is shown below:

```
corporation whitefield
```

The total cost of your shopped vegetables in the above test case is

```
60.00
```

Note that your final answer should have two decimal values. Also your program will be tested with different test files which will have different list but the format will remain the same. [HINT: You may wish to use strcmp to compare two strings from different files. Do man strcmp to know more about it.]

Task 4: mDonation (5 points)

IMTech2013 batch students are developing a mobile donation embedded systems in IIITB. This will help users select the donation in terms of different denomination such as Rs. 10, Rs. 5, Rs. 2 and Rs. 1. Write a program considering your code will be ported to their receiving embedded controller. Embedded system development represents using minimum number of variables in your code and thus minimizing the memory required to run program. You will have to choose the variable according to the size required. If the number of variables is more than the minimum required, your solution will not be evaluated.

Write a program in a file *donation.c* in *donation* directory. Your program receives the packet in an encoded version via command line argument and it has to decode the total value in each of the denomination. The data receiced in mDonation device is of following format:

```
|Ten(5-bit)|Five(3-bit)|Two(4-bits)|One(5-bits)|
```

The encoded data are provided in the following format:

```
./donation <data>
```

For example if your program is executed with the following example:

```
./donation 4641
```

then your program should print the following in integer format:

```
Following donation were made
Rs. 10
Rs. 5
Rs. 2
Rs. 1
```

You will get the above output because, 4641 represents 1 in each of denomination bit-space. 1 in 5-bit space of Rs. 10 represents Rs. 10.
1 in 3-bit space of Rs. 5 represents Rs. 5.
1 in 4-bit space of Rs. 2 represents Rs. 2.
1 in 5-bit space of Rs. 1 represents Rs. 1.

Another example is shown below:

```
./donation 4096
```

then your program should print the following in integer format:

```
Following donation were made
Rs. 10
```

Submitting your midterm exam

First, move into your `~/batch2` directory. Then run this system command:

```
    du -a
```

and check whether all your files are located in appropriate directory.

Once you are done and ready to submit, stand up and wait for your instructor to come to your place before submission. You need to submit your midterm exam only infront of the instructor. In case if you submit without instructors presence, your submission will be unaccepted. Also sign the attendance sheet with the time of submission before leaving the room. Submit your solutions while in the `~/batch2/` directory with the command:

```
    submit clab mr batch2 <your-iiitb.org-email-address>
```

Write your name, email-id and roll numbers in the question paper as well. Give your exam question paper back to your instructor. Have a nice break!!