CS 251: OutLab: Blast From The Past

- Due: 11:55 AM 1^{st} October
- Please write (only if true) the honor code. If you used any source (person or thing) explicitly state it. You can find the honor code on Piazza.

Overview

Lately there's been a huge rise in the number of superhero's {Bat,Spider,Ant........} man. All you need is an animal, then append the word man and BAM you have a new superhero. In fact there might come a time when there might be more superheros then known animals. I wonder what they'll do then. Anyways...., getting back on topic, you've decided to do take matters in your own hands and create a team of your own, to defeat the so called superheros. And what better place to find people to join your team than your past.

Note

In each of the task you'll obtain a string/image, which you need to mention in the text files corresponding to each task. The output will have a format that looks like

```
some text $characters/Item$ more text.
```

The characters or items (text between the two $ symbols) that you obtain in each of the tasks will be used as the input for (only) the last task. Depending on how many characters or items you collect the difficulty of the last task can vary.

A. Looney Tunes: Road Runner

Task

Ever wondered how Road Runner manages to pull of all those crazy stunts where he runs through wall but coyote ends up hitting the wall? In case you don't know what we are talking about check out this video.

The idea behind it is simple, he uses a super optimized materialization and de-materialization algorithm to convert walls into images and vice versa. But lately motivated by Bolt's performance in the Olympics, Coyote has grown much faster and hence Road Runner has enlisted your help in order to improve his materialization/de-materialization algorithm.

Road Runner's given you a copy of his beloved code for materialization/de-materialization in the file `RoadRunner/matDemat.cpp`. Using `gprof` figure out the slowest part of the code and optimize it.

In order to find the slowest part of the code consider using a smaller value of MAX in line 4 of the `matDemat.cpp` , and once you've done the optimization revert back to the original value of MAX, i.e., 1000.

Submission

In `TaskA.txt` mention how you optimized the code. Also submit the optimized version of `matDemat.cpp`

B. Beyblade

Task

Lately reverse rotation has been the new rage in the Beyblade universe. Though Tyson decided to take it one step further and he implemented what he likes to call "Intelligent Reverse Rotation", giving a bey the ability to change its direction in battle depending on the scenario. Awesome right!!.

Though after he installed the program in his bey, his bit beast, Dragoon found the program incompatible with it and it doesn't seem to compile properly.

You're given the `documented` code that implements "Intelligent Reverse Rotation" in `BeyBlade/intelligentRerverseSpin.cpp`. See if you can get it compiling and executing without any errors.

Submission

Submit the `doxygen configuration file` that you used for generating the html files and the working version of `intelligentReverseSpin.cpp`. In `TaskB.txt` mention how you figured out the changes that you needed to make to the code.

C. Power Puff Girls

Task

Sugar, Spice and Everything nice ....ohh and "Chemical X". Let's not forget "Chemical X" cause that's exactly what Professor Utonium is trying to synthesize today with Bubbles' help. Bubbles is tasked with controlling Dynamo (the giant robot) to mix some of the chemicals. Though after their recent battle there appears to be some bug in Dynamo. It keeps giving a segmentation fault for some reason.
The code that runs dynamo is given to you in `PowerPuffGirls/dynamo.cpp`. Try to get it running without any errors.

Suggested Reading

- gdb - backtrace
- gdb - frame

Submission

Submit the modified version of `dynamo.cpp`. In the file `TaskC.txt` mention how you used `gdb` to figure out the changes you made in `dynamo.cpp`

D. Scooby-Doo

Task

Scooby-Doo, Where are ~~You!~~ the soviet developed OZM-72 anti-personnel landmines.

Yeah..., Apparently Shaggy and the gang are stuck in front of a stretch of land plagued with land mines. Fortunately for them, the ever intelligent Velma was able to intercept the soviet transmissions and managed to get a list of integer coordinates. Though not all of the mines are active. The mines whose sum of coordinates is even are dead ones, and the other ones are live. For example

1. (89,10) - 99 - Active Mine
2. (12,12) - 24 - Inactive Mine

Obviously just looking at the numbers is not going to help them and hence the gang outsourced the job of coding a `python script` using `matplotlib` that takes in the co-ordinates of the points and generates a scatter plot. Though they want you to plot the live and dead mines in `two different colors`. The co-ordinates of the mines are given to you in `Scooby-Doo/Mines`

Submission

Submit the python script you've used for generating the points as `TaskD.py`. Also submit an image of the generated plot as `TaskD.png` or any other common image format.

E. Dexter's Laboratory

Task

You've probably heard about the Space X rocket explosion. Though what you've not heard about is why it exploded. What we are going to tell you now is TOP SECRET, so not a word to anyone.

The thing is Space X actually outsources its work to Dexter. The real genius.....and yes, it was his sister Dee Dee who played around with the code and messed it up. This resulted in all the pre flight checks failing. Dexter want's to check the output of the program and see if it would work if the flight checks passed.

Though unfortunately he only has the original executable and a fragment of the original source code. He's given you both of those in the `DextersLab` folder and asked you to find out what is the output of the original executable if all the pre flight checks pass.

Suggested Commands and Reading

- gdb set

Submission

In the file `TaskE.txt` mention all the steps you took to pass all the pre-flight checks.

F. Swat Kats

Task

The evil Dark Kat has once again sent a dangerous monster into the Megakat City and the Swat Cats T-Bone and Razor (we know them as Bade Miyan and Chote Miyan from Cartoon Network) are the only hope to save the city. Sadly their old weapons have not made any impact on the new deadly monster.

Fortunately, they devise a plan to defeat the monster and have created a new weapon with specialized ammo. But out of hurry they not labeled the ammo while creating them. Not all bullets have the same impact and the quicker bullets need to be fired first. You need to order the bullets to help the cats save the city. Each bullet is given as an executable in `SwatKats/`.

Suggested Commands

- time

Submissions

Submit the script `TaskF.sh` used to figure out the order of execution of the programs.

G. Power Rangers: Dino Thunder

Task

Lately as the monsters started to become more and more complex the time required for them to grow is increasing exponentially. In Fact the power rangers usually go for pizza after defeating the miniature version and still have plenty of time for assembling their megazords and defeating the huge versions.

On the bright side you've been threatened by Zeltrax to improve their monster growing algorithms. Looks like you have no option but to help him. The monster growing algorithm used by Zeltrax is given to you in `PowerRangers/`.

Submission

Submit the file that contains the optimized code. In the file taskG.txt mention the optimizations you made and why you made them.

H. The Final Battle - Extra Credit

Task

You've collected all the items and recruited all the members you could. It's finally time to thwart the eminent threat that the Earth faces from countless number of redundant super heroes.

The final battle takes place in the executable `FinalBattle/FinalBattle`. You start off by inputting all the characters you've recruited and all the items you've gotten. And then the battle starts. The battle takes place in a turn based format.

In each turn you can use one of your characters whose HP is not zero or one of the items that hasn't been used before to do some damage to an enemy character. You win if you can decrease the enemy's HP to zero.

Decrease you're enemy's HP to zero and win.

Suggested Commands

- gdb - info args
- gdb - info locals
- gdb - info variables
- gdb - info functions

Submission

There are multiple solutions to this question. Mention how you won the battle in `TaskH.txt`.

Submission Guidelines

1. When you submit, please document individual percentages such as Student 1: 80%, Student 2:100%, Student 3:10%. In this example, the second student will get full marks (10/10) and the first student will receive 8/10.
2. Do include a readme.txt (telling me whatever you want to tell me). Do include group members (name, roll number), group number, honour code, citations etc. This is the place for the reflection essay.

3. The folder and its compressed version should both be named `lab08_groupXY_outlab` for example folder should be named `lab08_group07_outlab` and the related `tar.gz` should be named `lab08_group07_outlab.tar.gz`

4. The folder you submit should have the following structure. Note that there's a separate folder for each task

```
lab08_groupXY_outlab
├── TaskA
│   ├── TaskA.txt
│   └── matDemat.cpp
├── TaskB
│   ├── TaskB.txt
│   ├── Doxyfile
│   └── intelligentRerverseSpin.cpp
├── TaskC
│   ├── TaskC.txt
│   └── dynamo.cpp
├── TaskD
│   ├── TaskD.py
│   └── TaskD.png or TaskD.jpg or ..
├── TaskE
│   └── TaskE.txt
├── TaskF
│   └── TaskF.sh
├── TaskG
│   ├── TaskG.txt
│   └── Optimized monster growing file
└── TaskH
    └── TaskH.txt
```

How We will Grade You [70 + 8 Marks]

In the txt file of each task mention the string that you obtain after solving the question. This accounts for 1 Mark for each of the tasks.

1. Task A [**9 Marks**]

2. Task B [**11 Marks**]

3. Task C [**11 Marks**]

4. Task D [**9 Marks**]

5. Task E [**12 Marks**]

6. Task F [**9 Marks**]

7. Task G [**9 Marks**]

8. Task H [**8 Marks**]