gdb Practice Exercise; Project 1 Turn-in

CS 411

gdb Debugging Exercise

In addition to using gdb to debug a program, this exercise will also give you some practice creating a git/GitHub repository from scratch. I'd recommend using phoenix for this exercise.

```
# You're on another system/different account, so you'll need to repeat
# the following tooling configuration.  Re-use your identity values.

git config --global user.name "John Doe"
git config --global user.email JohnDoe@example.com
git config --global push.default simple
git config --global core.editor "emacs -nw" # or just emacs or gedit or vi

# Again, run one of these two commands:

#  1) Cache credentials in memory for one hour (units are seconds).  Adjust
#  the timeout as you see fit.

git config --global credential.helper "cache --timeout=3600"

#  2) Store credentials unencrypted on disk permanently
#     Default storage file is ~/.git-credentials

git config --global credential.helper store

# First, create a new team and GitHub repository for this exercise.
#
# If you haven't already done so, go into GoucherLearn and follow the link
# you'll find there to create your team and your team's GitHub repository.
# This repository is <new-repository> below.  Follow this rule when you
# create your team name:
#    <initials of 1st member>_<initials of 2nd member>_1_FIXME
# Example:
#    JLZ_TPK_1_FIXME

# One member of your team should run the next several git commands, up to
# and including the 'git push -u origin master' command.

# Create a directory for your gdb exercise.  Download fixme.c from the
# class web site and store it in this directory.  While you're at it,
# create README.md in the directory and add a few lines to it.  Now, from
# inside this directory, initialize a new git local repository, add the two
files to be staged for the initial commit, and make the initial commit:

git init
git add .
git commit -m "Initial commit."

# Now, add your remote, empty GitHub repository.  For ease of reference, it
# will have "origin" as its name.  (Otherwise, you'll be typing the full URL
# every time you do a push or a pull.)

git remote add origin <new-repository>.git

# Now, push your local repository to the remote, setting the default for
# upstream to be the master branch of the origin remote repository:

git push -u origin master

# From this point on, you, and team members who have cloned this repo, can
# push to the remote using just

git push

# and pull from the remote using

git pull

# Other team members can now clone the repo:

git clone <new-repository>.git
```

Carefully read the comments in `fixme.c` Using gdb, find and fix each of the five bugs in the program. As you fix each bug, leave brief comments in the program itself. In `README.md`, explain the nature of each bug and how you fixed it.

Project 1 Turn-in

By the project deadline, email to me the GitHub https URLs of your kernel repository, and your gdb exercise repository. The names of your team members should be listed in the README.md file in each of the repositories. Make sure that the work that you want me to assess is in the (default) master branch of your repositories. For example, I will be using the following sequence to pull your kernel syscall work and set the branch that I will assess from my local copy of the kernel repo:

```
git remote add foo <your-repository>.git
git fetch foo
git checkout -b foo foo/master
```

This project will be assessed as follows:

- Kernel compile/syscall exercise: 30%
- gdb exercise: 60%
- Documentation: 10%

Remember to document assistance you received from others. This can be done in the email you send me your repo URLs, in your README.md files, or in comments at the very top of your source code files (see fixme.c for an example).