Lab2 – Goals

• Practice program optimization workflow

• Gain understanding & practical experience in
  • Optimizing program for memory performance
  • Using profiling tools + inspection tools
  • Optimizations performed by the compiler
  • Man page reading comprehension and googling skills
Lab2 – Overview

• **Time Frame:** Sept 24 – Oct 18 (~3 week)

• **Lab Marks:** Worth 9% of total course mark
  - 30% competitive, 70% non-competitive (minimum 150x speedup)
  - Competitive mark calculated on log base 2 scale

• **Team Composition:** Individual work

• **My Suggestion:**
  
  Do not perform pre-mature program optimizations
  
  **Do spend** lots of time on checking out what compiler can do behind your back
  - Pretend you are the compiler and optimize for algorithm + code
  
  **Do spend** more time reading, thinking, profiling rather than writing code
  - “Genius is 2% inspiration, 98% perspiration” - Thomas Edison
Lab2 – Overview

Start with a square image

• White pixel = background
• Non-white pixel = object
Lab2 – Overview

Given a series of inputs
- WASD, Rotate CW+CCW, Mirror X+Y
- Manipulate object inside the image
- Output result image every 25 input instructions

Move down by 10 pixels
Lab2 – Overview

Given a series of inputs

• WASD, Rotate CW+CCW, Mirror X+Y
• Manipulate object inside the image
• Output result image every 25 input instructions
Lab2 – Overview

Given a series of inputs

• WASD, Rotate CW+CCW, Mirror X+Y
• Manipulate object inside the image
• Output result image every 25 input instructions
Lab2 – Overview

Given a series of inputs
- WASD, Rotate CW+CCW, Mirror X+Y
- Manipulate object inside the image
- Output result image every 25 input instructions
Lab2 – Optimization Headroom

I have tried very hard to make the reference solution slow

Level of difficulty:

• You should easily achieve 150x+ speedup with modest effort in the right places
• One student already achieved closed to 500x speedup

Really big hints on how to get high marks:

• Think hard on why the reference solution is slow
• Compiler optimization techniques can be used in your algorithms as well
• Don’t perform premature optimization (readability is important)
• Improve your solution and lower other’s competitive marks
Lab2 – Optimization Workflow

- Evaluate current algorithm
- Optimize your algorithm
- Profile your code
- Optimize for system performance

**Give up because**
- Your time is better spent elsewhere such as studying quizzes
- You are too good and you are afraid others will beat you
  - Warning to 1st place student: TA is not responsible for your personal safety
Lab2 – Am I cheating?

When in doubt, you can ask the TAs

I usually wouldn’t consider you cheating if:

• You can explain clearly why you wrote the piece of code
• You can explain clearly why a piece of code works and how it works
• We can get consensus from top students that it’s a legit method

Obvious cheating examples:

• Encode results into the source code to avoid doing computation
• Caching previous execution attempts and avoid doing actual work
• Submitting other people’s code or part of people’s code
  • We received 100+ source code versions from this year’s class already
  • We keep all versions of submissions in the last 4 years
Helpful Resource

GCC Man Page: https://lmgtfy.com/?q=gcc+man+page
Gprof Tutorial: https://lmgtfy.com/?q=gprof+tutorialspoint
Gcov Tutorial: https://lmgtfy.com/?q=gcov+tutorialspoint
Git tutorial: https://lmgtfy.com/?=git+tutorial+for+beginners
Bash Script Tutorial: https://www.shellscript.sh/