To print higher-resolution math symbols, click the
Hi-Res Fonts for Printing button on the jsMath control panel.

Lab 8 Grading Guidelines last modified by Aaron Bloomfield on August 13, 2013 11:27:16 AM EDT

Lab 8 grading guidelines

Pre-lab

Out of 10 points. This lab part **is** expected to compile.

- 5 points: multiplication function
  - 3 points: output
    - 1 point for each correct output (the first of the two numbers printed)
  - 2 points for the quality of code
    - full credit if the function works
    - -2 (i.e. no credit for this sub-part) if the does not work correctly
  - In addition, the following penalties apply:
    - -3 if it was not *iterative*
    - -5 (i.e. no credit) if it uses the imul (or similar) opcode - they should only be using addition
- 5 points: power function
  - 3 points: output
    - 1 point for each correct output (the second of the two numbers printed)
  - 2 points for the quality of code
    - full credit if the function works
    - -2 (i.e. no credit for this sub-part) if the function does not work correctly
  - In addition, the following penalties apply:
    - -3 if it was not *recursive*
    - -5 (i.e. no credit) if it uses a power (or similar) opcode - they should only be using addition
    - -3 if they do not call *their* multiplication routine, but call imul (or similar) instead

- 4 points : Did not allow for input.
- 2 points: Wrong/No output but they put effort into their assembly code (1 point for each function)
- If they switch their iterative/recursive functions, then take off 1 point for each. The multiplication was supposed to be iterative, and power recursive; if they reversed this, then they get the 1 point off for each part.

Pre-lab input

They are supplying their own mathfun.cpp file, so there could be several differences in output and possibly in the way they are getting input.

```
1st Trial Parameters passed in: 2 5
2nd Trial Parameters passed in: 5 3
3rd Trial Parameters passed in: 3 4
```

Pre-lab output

For their output, we are just looking for the correct numbers; we don't care about the presentation of said numbers.

```
The parameters passed in are 2 and 5.
The product of 2 and 5 is: 10.
2 raised to the 5 is: 32.

The parameters passed in are 5 and 3.
The product of 5 and 3 is 15.
5 raised to the 3 is: 125.
```

```
The parameters passed in are 3 and 4.
The product of 3 and 4 is 12.
3 raised to the 4 is: 81.
```

In-lab

Out of 10 points. This lab part is **not** expected to compile.

Either Parameter Passing or Objects could be chosen for the in-lab. We are mainly looking to see if they hit *most* of the main points in one of the topics. They should have a rough write-up of the material, showing that they did some work during the in-lab. This is the spirit of what we are looking for. Feel free to give them a 10 if they appear to fulfill the spirit of this part of the lab.

- Had to complete 2 parts of one section, Parameter passing 1, 2, or 3 or Objects 1, 2, 3, or 4
- 6 points are based off understanding, much as the points below.
    - 6 points for hitting all major points
    - 4 points for showing some understanding but missing a few key steps
    - 2 points for putting something close to coherent, showed work had been put in
    - 0 for nothing
- 2 points were determined by whether the student included the assembly code they generated from their work
- 2 points were determined by grammar, organization, following directions, etc.

Post-lab

Out of 10 points. This lab part is **not** expected to compile.

The lab report must cover all of the key points described in the Lab 8: x86, part 1 page for Parameter passing and Objects. The grading will be based on the quality and content of the report.

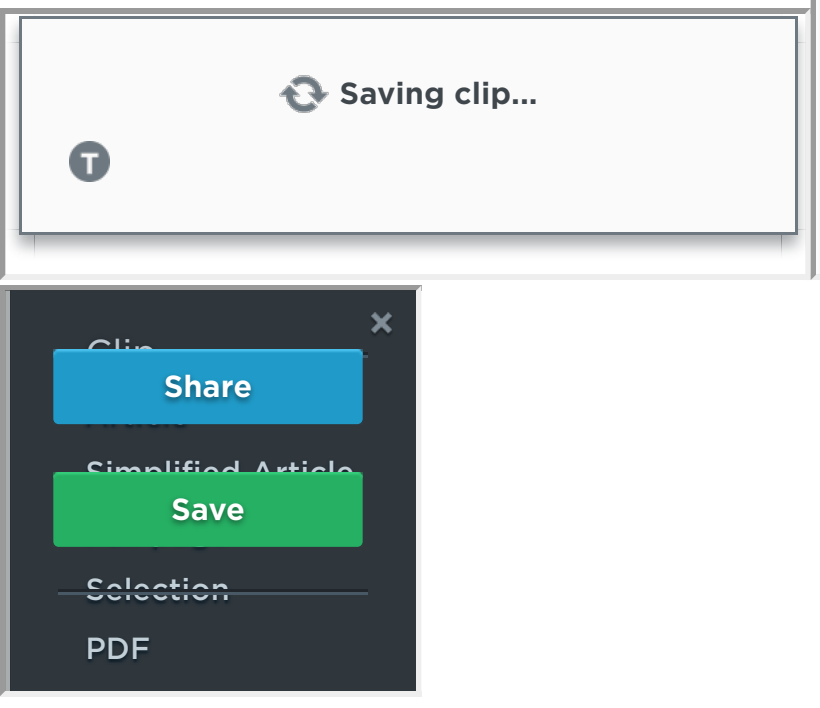
- 2 points (1 point for each source up to 2): At least 2 sources were cited
- 2 points (1 point for each supporting type of information): There is strong evidence given supporting their report in at least 2 of the following ways (they can have other ways as well - these are just examples):
    - Code snippets
    - Screen shots
    - Citations from outside sources
- 3 points: Analysis and Hypotheses
    - 3 points: Everything was well explained and educated hypotheses were made when there were things unknown.
    - 2 points: Not everything is explained nor even addressed in the analysis.
    - 1 point: Poor explanations, and little attempt was made at making hypotheses when they did not know what was going on.
- 3 points: Key points for Parameter Passing and Objects were covered
    - 3 points: Everything seems to be covered, and covered well
    - 2 points: Hit most of the topics, but missed a few here and there
    - 1 points: There were a lot of things missing
    - 0 point: Poor quality job, lots of topics missing, etc.

Key Points for Parameter Passing

- Part 1. (5 points - 1 point per data type)
    - Explains how the following are passed by value and by reference:
        - int
        - char
        - pointers
        - floats
        - objects with more than one data member
- Part 2. (3 points - 1 point per explanation)
    - Explains how arrays are passed in C++
        - Explains how values are passed into a function
        - Explains how callee accesses parameters inside of the function
        - Explains where data is placed in terms of at least a register-relative address if not actual address
- 2 points for quality of the report as in grammar, organization, etc.

Key Points for Objects

- Part 1. (2 points - 2 points for covering everything, 1 point if incomplete)
    - Explains how the following works in C++ objects:
        - Data layout - how data is kept in memory
        - Data member access - how assembly knows which data member to access
        - Method invocation - how assembly knows which object it is being called out of
- Part 2. (2 points - 2 points for covering everything, 1 point if incomplete)
    - Explains how data is laid out specifically for a C++ class (note before they could have used a struct, but here it has to be a class)
        - At least 5 data members were included in the class that they are using
        - Different data types were explored
        - Different access levels (protected, private, public) were investigated
- Part 3. (2 points - 2 points for covering everything, 1 point if incomplete)
    - Explains how data members are accessed both from inside a member function and from outside (the actions performed in assembly to access a member function)
- Part 4. (2 points - 2 points for covering everything, 1 point if missing either how public member functions are accessed or how the "this" pointer is implemented)
    - Explains how public member functions are accessed for their sample class.
    - Explains how the "this" pointer is implemented. They should address most of the questions below:
        - Explains where it is stored
        - Explains when it is accessed
        - Explains how it is passed to member functions
        - Explains when it is updated
- 2 Points for quality of the report as in grammar, organization, etc.


Saving clip...
To:
Share
Save
Selection
PDF