Basics2 – Overloading C++

Due Date

- See Piazza for due date and time
  - Grading the next day
- Submit program to perforce in your student directory
  - Sub directory called:
    - /Basics2/...
  - Fill out your ***Basics2 Submission Report.pdf***
    - Place it in the same directory as your solution
    - Enter the final Changelist number of your submission
    - Enter the number of test passed
    - Write up a quick discussion in the report
      - What you learned from this basics

Goals

- C++ Proficiency
  - Real-World Overloading
- Increasing C++ knowledge and understanding

Assignments

- General:
  - Add methods and operators for overloading.
  - Run the Unit Tests to verify progress / success
    - 12/12 is the best for this program

- Monkey Class - Description

  - Background
    - Monkey class and Nibble class will be modified to support overloading correctly.
      - Monkey class - adding the Big Four operators
      - Nibble class - adding Big Four + various overload operators
    - The unit tests shake out the program and verify the correct functionality
  - private:
    - Monkey has 2 private variables, x and y
    - Monkey has one char string pointer called status.
  - public:
    - There are several public methods supplied
      - getX(), getY(), getStatus(), printStatus()

- Methods to Add
  - The Big Four operators to public methods
    - Default constructor
      - initialize
        - x: 111
        - y: 222
      - Dynamically create (use new) char string, *status*.
        - initialize to: "Initialized with default"
    - Copy constructor
      - deep copies the string
    - Assignment operator
      - deep copies the string
    - Destructor operator
      - deletes the *status* char string
      - use delete
  - Specialize constructor
    - Initialize variables x and y with the passed parameters
    - Dynamically create (use new) char string, *status*.
      - initialize to: " "Initialized by user""
- Update the two static variables where appropriate
  - For every new allocation of a string increment
    - numStringsCreated
  - For every deletion of a string increment
    - numStringsDestroyed
  - See unit tests for verification
    - Reverse engineer the test functions for examples and clarity
- Modify the Monkey class and run the unit tests

- *Nibble Class* - Description

  - Background
    - This is class creates an abstract data type, Nibble (4 bits)
      - With overloaded operators
    - You can add numbers to this data type, it will wrap if it exceeds the 4 bits of storage.
  - private:
    - Storage of the 4 bit data (actually its 8, but we are treating it as 4 bit)
  - public:
    - Method getData() returns the data
  - Methods to Add
    - The Big Four operators to public methods
      - Default constructor
      - Copy constructor
      - Assignment operator
      - Destructor operator

- Binary operators
  - Nibble + constant
  - constant + Nibble
  - Nibble + Nibble
  - Nibble += Nibble
- Unary operators
  - ~ operator
    - Ones complement
  - +operator
    - returns the positive value
  - casting operator() to an unsigned int
    - Adds 5 to the value (for academic purposes)
  - pre-increment ++
    - ++Nibble
  - post-increment ++
    - Nibble++
  - operator <<
    - Use as a rotational shift function within the nibble
    - Each bit rotates to the left by the number specified
    - If a bit fall off the edge it is rotated to the beginning bit.
      - x: 1110b x<<1 answer x: 1101b

- Modify the Nibble class and run the unit tests

- Check in the problems multiple times, at least 3 times for this PA (programming assignment)
  - Have reasonable check-in comments
- Make sure that your program compiles and runs
  - Warning level 4 sometimes that is not possible due to MS headers…
  - Your code should be squeaky clean.
- Submit program to perforce in your student directory
  - Sub directory called: /Basics2/…

Validation

Simple check list to make sure that everything is checked in correctly

- Did you do all run all unit tests problems?
- Do they compile and run without any errors?
- Warning level 4 free?
- Submitted it into /Basics2 directory - without the extra files?
- Submit the submission report?
- Can you delete you local drive, regrab the Basics2 directory?
    - Is all the code there?

Hints

Most assignments will have hints in a section like this.

- This is pretty easy Basic assignment
- I expect this assignment to be completed quickly for most of the students
    - Hardest part was creating these tasks and writing the Unit Tests
- You will initially get complier overload
    - Disable files from build
        - Slowly fix and enable files
    - Comment out unit tests within one file
        - Work on one test at a time slowly fixing individual issues.
        - Comment out tests within files or use #if 0 trick