$Id: cmps109-2015q2-exam2.mm,v 1.66 2015-05-12 16:20:40-07 - - $

page 1 page 2 page 3 Total / 32

Please print clearly :

Name :

Login : @ucsc.edu

Code only in C++11. No books ; No calculator ; No computer ; No email ; No internet ; No notes ; No phone. Neatness counts ! Do your scratch work elsewhere and enter only your final answer into the spaces provided.
For all answers, assume : `using namespace std;`

1. Complete the following declarations of primary and secondary colors. Use either `0x00` or `0xFF` in your answers. (Points : 0 if none or one correct ; 1/2 if two or three correct ; 1 if four correct.) **[1✔]**

`const GLubyte RED[] = {0x ___, 0x___, 0x___};`	`const GLubyte GREEN[] = {0x ___, 0x___, 0x___};`
`const GLubyte BLUE[] = {0x ___, 0x___, 0x___};`	`const GLubyte CYAN[] = {0x ___, 0x___, 0x___};`

2. A polynomial can be respresented as a vector, with coefficients stored at the position given by their power. For example, $8x^5+9x^3+7x^2+10$ would be represented as the vector `{10, 0, 7, 9, 0, 8}`. That is, the term ax^n is stored in a vector `v` with `v[n]==a`. The partial header file `polynomial.h` is shown here. Write code that would be placed in the implementation file `polynomial.cpp`.

```
class polynomial {
   friend ostream& operator<< (ostream&, const polynomial&);
   private: vector<double> v;
   public: bool operator== (const polynomial&) const;
           polynomial& operator+= (const polynomial&);
}
polynomial operator+ (const polynomial&, const polynomial&);
```

   (a) Code `operator==`. **[1✔]**

   (b) Code `operator+=`. **[3✔]**

   (c) Code `operator+` (which is neither member nor friend). **[1✔]**

   (d) Code `operator<<`. Example : The vector $9x^4+8x^2+10$ should be printed as `9 x4 + 8 x2 + 10`, and $9x^8+11x+3$ should be printed as `9 x8 + 11 x + 3`. Coefficients having the value 0 are not printed. If all coefficients are 0, print `0`. **[4✔]**

3. Complete the following operators, assuming `operator==` and `operator<` are defined. **[2✔]**

```
template <class T>
inline bool operator!= (const T& x, const T& y) {                    }
```

```
template <class T>
inline bool operator>  (const T& x, const T& y) {                    }
```

```
template <class T>
inline bool operator<= (const T& x, const T& y) {                    }
```

```
template <class T>
inline bool operator>= (const T& x, const T& y) {                    }
```

4. Code the template function `accumulate`. It takes two iterators and a function object as arguments. It applies the function object to each of the elements in the range indicated by the iterators. **[1✔]**

```
template <typename iterator, typename funobj>
void accumulate (iterator begin, iterator end, funobj& fun) {



}
```

5. Define a struct `adder` which might be passed into the function above. It has an `operator()`, which when applied to any elements in the range, adds the argument to a `double sum` member of the struct. It might be called by: `adder a; accumulate (v.begin(), v.end(), a);` where `v` is a container of doubles. **[1✔]**

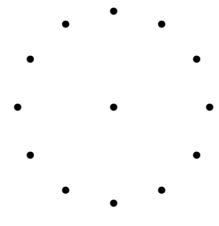
```
struct adder {



};
```

6. Write a function `remove` which takes a container and a predicate and erases all elements of the container for which the predicate returns true. The member function `erase` accepts an iterator into the container as an argument, removes the element and returns an iterator pointing at the next element. **[2✔]**

```
template <typename container, typename T>
void remove (container& c, bool(*p)(T)) {







}
```

7. Using OpenGL, complete the function `draw_dots`, which draws dots at the 12 hours as points on a clock, and at the center of the unit circle. The function call `glVertex2f(x,y)` specifies the position of the dot, where `x` and `y` are of type `float`. The center of the circle of dots is at $(0, 0)$ and the circle has unit radius. See the diagram at the right. Assume `#include <cmath>`, which defines `M_PI`, `sin`, and `cos`. **[2✔]**

```
void draw_dots() {
   glBegin(GL_POINTS);









   glEnd();
}
```



8. Write a template function that takes a container, a predicate, and an action function, and applies the action function to each element of the container, for which the predicate is true. **[2✔]**

```
template <typename container, typename predicate, typename action>
void apply (container& c, predicate p, action a) {
```

Multiple choice. To the ***left*** of each question, write the letter that indicates your answer. Write ***Z*** if you don't want to risk a wrong answer. Wrong answers are worth negative points. **[12✔]**

number of correct answers		× 1 =	$= a$
number of wrong answers		× ½ =	$= b$
number of missing answers		× 0 =	0
column total $c = \max(a - b, 0)$	12		$= c$

1. In the listmap project, how many pointers are there in the data structure declared as `listmap<int,int>` if n pairs of `int`s are in the data structure ?
   (A) n
   (B) $n+1$
   (C) $2n$
   (D) $2n+2$

2. Which C++98 library element is deprecated in C++11 ?
   (A) `T*`
   (B) `auto_ptr<T>`
   (C) `shared_ptr<T>`
   (D) `unique_ptr<T>`

3. Which of the following is not a standard cast in C++11 ?
   (A) `const_cast`
   (B) `dynamic_cast`
   (C) `lexical_cast`
   (D) `reinterpret_cast`

4. A function dynamically dispatched (chosen at runtime, not compile time) is indicated by which keyword ?
   (A) `friend`
   (B) `public`
   (C) `template`
   (D) `virtual`

5. How is an abstract member function declared ?
   (A) `abstract virtual f(void);`
   (B) `abstract void f();`
   (C) `virtual void f() = 0;`
   (D) `virtual void f() = abstract;`

6. If a class has a virtual function declared, what must also be declared virtual ?
   (A) `copy constructor`
   (B) `copy operator=`
   (C) `default constructor`
   (D) `destructor`

7. The `vector<T>::iterator_category` is :
   (A) `bidirectional_iterator_tag`
   (B) `forward_iterator_tag`
   (C) `input_iterator_tag`
   (D) `random_access_iterator_tag`

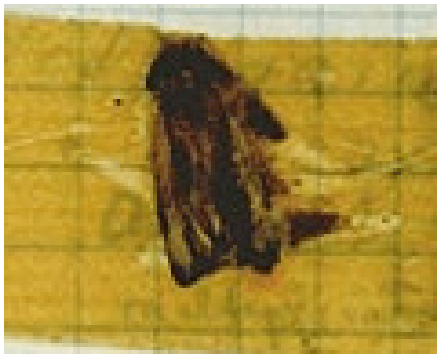
8. How does `shared_ptr<T>` manage memory ?
   (A) concurrent multithreads
   (B) copying collector with semispaces
   (C) mark and sweep
   (D) reference counting

9. If `object` is the root of an object-oriented hierarchy, which of the following should be marked `= delete` ?
   (A) bool operator==
   (B) copy constructor
   (C) default constructor
   (D) destructor

10. For class `map<string,node_ptr>`, What is `node_ptr` ?
    (A) `map<string,node_ptr>::allocator_type`
    (B) `map<string,node_ptr>::key_type`
    (C) `map<string,node_ptr>::mapped_type`
    (D) `map<string,node_ptr>::value_type`

11. What is the amortized efficiency of a search through a `std::map` ?
    (A) $O(1)$
    (B) $O(\log_2 n)$
    (C) $O(n)$
    (D) $O(n \log_2 n)$

12. If `c` is a `list<int>` What will iterate over the list ? Fill in the blank.
    `for (auto i = c.begin(); _____; ++i) f(*i);`
    (A) `i != c.end()`
    (B) `i < c.end()`
    (C) `i == c.end()`
    (D) `i > c.end()`



The First "Computer Bug". Moth found trapped between points at Relay #70, Panel F, of the Mark II Aiken Relay Calculator while it was being tested at Harvard University, 9 September 1947. The operators affixed the moth to the computer log, with the entry: "First actual case of bug being found." They put out the word that they had "debugged" the machine, thus introducing the term "debugging a computer program". In 1988, the log, with the moth still taped by the entry, was in the Naval Surface Warfare Center Computer Museum at Dahlgren, Virginia. [`http://en.wikipedia.org/wiki/File:H96566k.jpg`]