$Id: cse111-2019q4-final.mm,v 1.120 2019-12-04 16:15:28-08 - - $

page 1	page 2	page 3	page 4	page 5	Total / 54

Last Name :	
First Name :	
CruzID :	`@ucsc.edu`

No books ; No calculator ; No computer ; No email ; No internet ; No notes ; No phone. Neatness counts ! Points will be deducted for messy or unreadable answers. Do your scratch work elsewhere and enter ONLY your final answer into the spaces provided, and ONLY in the spaces provided.

1. Inheritance and virtual functions. All pointers in this question are to be raw pointers (`expr*`). Code all function bodies inline. Make proper use of the keywords `virtual` and `override`. Declare functions as `const` and/or abstract when appropriate.
   `class null_tree_error: public runtime_error {};`

   (a) Define an abstract base class called `expr`. **[4✔]**
   - (i) `eval` has no arguments and returns a `double`.
   - (ii) `print` returns `void` and has an `ostream` argument.
   - (iii) `default` the default constructor and destructor.
   - (iv) `delete` the copiers and the movers.

   (b) Class `leaf` inherits from `expr` and has a `double` value field. **[2✔]**
   - (i) `eval` just returns this value and `print` just prints it.
   - (ii) The constructor has one optional argument which defaults to 0.

   (c) Class `tree` inherits from `expr` and has two raw pointers to `expr`s (`left` and `right`).
   - (i) `eval` returns the sum of the values of the two subtrees. **[4✔]**
   - (ii) `print` prints the two subtrees recursively bounded by a pair of parentheses and connected by a plus signs. Example : `((6+8)+(9+44))`.
   - (iii) `null_tree_error` is thrown by `eval` and `print` if either subtree is `nullptr`.
   - (iv) The constructor takes two arguments (the left and right subtrees).
   - (v) The destructor prevents memory leak.

2. Define template `operator<<` in the standard way so that `cout<<cont` for any container will print out the contents of the container: print an open brace (`{`) then each element of the container, then a close brace (`}`). If there is more than one element in the container, print commas between elements, but no comma in before the first element or after the last element. **[2✔]**

3. Define a template function `palindrome`, which takes any container and returns true if its elements constitute a palindrome, and false otherwise. Assume the container provides bidirectional iterators, which means that they have both `operator++` and `operator--`. A palindrome is any sequence which is identical in either a forward a reverse direction. Examples: `string("ablewasiereisawelba")`, `{'a','b','c','b','a'}`, `{1,2,3,2,1}`, `{1}`, `{}`, `{1,2,2,1}`, `{"foo","foo"}`, are all palindromes, but `{1,2,3,4,2}` is not. **[3✔]**

4. Write a template function called `test_palindrome` which takes any container and calls `palindrome`. It then prints the `true` or `false` result of this call, as words, not as 0 or 1, and after that prints all of the elements of the container, each element preceded by a space. Use the colon version of the `for`-loop. Print a newline at the end of the sequence. **[2✔]**

5. Given `ubigint` as in the project, define `ubigint::operator<` as it would appear in the implementation file (not inside the class definition). **[3✔]**

```
class ubigint {
      vector<unsigned char> uvalue;
   public:
      bool operator< (const ubigint&);
```

6. Write a **bash** shell script which will iterate over all test files matching the pattern **test*.ydc**, and run the program **ydc**, redirecting the standard input from that input file, and redirecting the standard output to a file with the same name as the input file, except suffixed with **.out**. (Example: if the input file is **test3.ydc**, then the output file is **test3.ydc.out**). **[1✔]**

7. Polymorphism: In each box, write **U** for universal, **A** for ad-hoc. Also, write **C** for conversion, **O** for overloading, **P** for parametric, **I** for inclusion. Score: 1 ✔ if 4 correct; ½ ✔ if 2 or 3 correct; else 0. **[1✔]**

`void f(int);` `void f(string);`	`class baz: private qux {` `};`
`template <typename T>` `T sum (T*);`	`void f(double);` `int x; f(x);`

8. Define the inline template **operator>** which uses **operator<**. **[1✔]**

9. Assume that **foo::operator++()** has been defined as a member prefix operator. Define the postfix **operator++** as a non-member. Use the prefix operator in the definition. **[1✔]**

10. Assume a class **iterator** contains a pointer to a **foo**. Define an inline member of **iterator** which allows it to be implicitly converted to a **bool**. **[1✔]**

```
class iterator {
      foo* pointer;
   public: ____________________________________________________
```

11. Assume that **bigint::operator+=** has been defined already. Define a non-member **operator+** which takes two bigint arguments and returns the sum. **[1✔]**

12. Define a template inline function **max** which takes two arbitrary arguments and returns the larger one. Assume **operator<** has been defined for the arguments. **[2✔]**

13. Write a portion of a **Makefile** which will build an executable binary from object files and build object files from source files. The first target should be **all**. Assume the following **make** variables have been defined: **[2✔]**

EXECBIN = the name of the executable binary.
OBJECTS = the variable containing the list of object files.
GPP = the compilation command with appropriate options.

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. What is the type of `map<string,double>::value_type` ?
   (A) `pair<const string, const double>`
   (B) `pair<const string, double>`
   (C) `pair<string, const double>`
   (D) `pair<string, double>`

2. Given an iterator pointing somewhere into the middle of one of these collections, what is the running time required to delete the element at which it points ?
   (A) `vector` is $O(1)$; `list` is $O(1)$
   (B) `vector` is $O(1)$; `list` is $O(n)$
   (C) `vector` is $O(n)$; `list` is $O(1)$
   (D) `vector` is $O(n)$; `list` is $O(n)$

3. What cast can convert a pointer to a `uintptr_t` ?
   (A) `const_cast`
   (B) `dynamic_cast`
   (C) `reinterpret_cast`
   (D) `static_cast`

4. Which system call will allow a process to communicate with another process running on a possibly different computer ?
   (A) `fork`(2)
   (B) `pipe`(2)
   (C) `popen`(3)
   (D) `socket`(2)

5. What `Makefile` variable should fill in the blank in the following recipe ?
```
${EXECBIN} : ${OBJECTS}
        ${GPP} -o ____ ${OBJECTS}
```
   (A) `$<`
   (B) `$@`
   (C) `$Id$`
   (D) `%.o`

6. If a group of classes are designed to cooperate in such a way that when a function using an expression like `p->f(x)`, the actual function being called is determined at run time, depending on the class of `p`, then `f` should be declared with what attribute ?
   (A) `const`
   (B) `friend`
   (C) `template`
   (D) `virtual`

7. What function member of class `foo` will allow a `foo` object to be used in a boolean context (such as an `if`-statement) ?
   (A) `bool operator() const;`
   (B) `foo (bool);`
   (C) `friend operator<< (ostream&, bool);`
   (D) `operator bool() const;`

8. Which operator can be declared with zero, one, two, three, or even more parameters ?
   (A) `operator()`
   (B) `operator+`
   (C) `operator++`
   (D) `operator[]`

9. What is the expected prototype for `operator<<` that will print a `foo` ?
   (A) `foo& operator<< (ostream&, const foo&);`
   (B) `ostream& operator<< (const ostream&, foo&);`
   (C) `ostream& operator<< (ostream&, const foo&);`
   (D) `ostream& operator<< (ostream&, foo);`

10. The status result of `waitpid`(2) is a 16-bit number. If a program crashes, the low-order 7 bits contain the signal number. What expression will extract the signal number ?
    (A) `signal & 0x7F`
    (B) `signal ^ 0x7F`
    (C) `signal | 0x7F`
    (D) `signal >> 0x7f`

11. Which keywords will prohibit a constructor from being used as an implicit conversion ?
    (A) `explicit`
    (B) `friend`
    (C) `implicit`
    (D) `private`

12. Is half of two plus two equal to two or three ?
    (A) no
    (B) three
    (C) two
    (D) yes

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. Given the following declaration :
   `union { int a; unsigned char c[4]; };`
   if `a` is `0x12345678`, and assuming an `int` is 4 bytes, then what is `c[0]` on a little-endian machine ?
   (A) `0x12`
   (B) `0x34`
   (C) `0x56`
   (D) `0x78`

2. Failure to `delete` variables on the heap that have been allocated with `new` will cause :
   (A) assertion failure
   (B) dangling pointers
   (C) memory leak
   (D) segmentation fault

3. The return type of function `main` is required to be :
   (A) `char**`
   (B) `int`
   (C) `void`
   (D) any of the above is OK.

4. What does `map::find` do if it fails to find ?
   (A) `return map::begin() - 1`
   (B) `return map::end()`
   (C) `throw map::find::not_found`
   (D) `throw std::out_of_range`

5. The first non-comment line in `foo.h` should be :
   (A) `#define __FOO_H__`
   (B) `#ifdef __FOO_H__`
   (C) `#ifndef __FOO_H__`
   (D) `#include <foo.h>`

6. Assuming the container `v` allows random access iterators, which of the following is true ?
   (A) `v.size() == v.begin() - v.end()`
   (B) `v.size() == v.begin() - v.end() + 1`
   (C) `v.size() == v.end() - v.begin()`
   (D) `v.size() == v.end() - v.begin() + 1`

7. Which of the following iterators allows `--i` and `i--`, but not `i = i - 1` ?
   (A) input
   (B) forward
   (C) bidirectional
   (D) random access

8. When a server is ready for a client to connect, which system call will cause it to wait for a client to connect ?
   (A) `accept`(2)
   (B) `bind`(2)
   (C) `listen`(2)
   (D) `socket`(2)

9. What color does the following define ?
   `const GLubyte ____[] {0xFF, 0xFF, 0x00};`
   (A) `BLUE`
   (B) `MAGENTA`
   (C) `RED`
   (D) `YELLOW`

10. Given `map<string,double> m;`, after the statement `p = m.find("foo");`, what expressions will produce the associated number ?
    (A) `p->first`
    (B) `p->second`
    (C) `p.first`
    (D) `p.second`

11. Assuming the usual semantics, what expression is equivalent to `a == b` ?
    (A) `(a < b) and (b < a)`
    (B) `(a < b) or (b < a)`
    (C) `not (a < b) and not (b < a)`
    (D) `not (a < b) or not (b < a)`

12. A process that waits in the background doing nothing except wait for a client process, and then wakes up to perform a service is called a :
    (A) daemon
    (B) vampire
    (C) werewolf
    (D) zombie