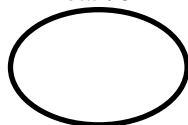


Total / 32



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! Points will be deducted for messy or unreadable answers. Do your scratch work elsewhere and enter only your final answer into the spaces provided.*

1. Given an outline for a container shown here, write prototypes for the functions on both the container and its iterator that are used by the following statement. Do now show function bodies. [3✓]

```
thing t; for (auto& i: t) cout << i << endl;
```

```
class thing {
    class iterator;

};
```

```
class thing::iterator {

};
```

2. Rewrite the **for**-statement using the two semi-colon version of a **for**-loop but not the colon. Explicitly use the iterator. [1✓]

```
thing t; for (auto& i: t) cout << i << endl;
```

3. Write a function to compute the inner product of two `vector<double>`. If the vectors are of different lengths, throw a `domain_error`. The formula for inner product of vectors  $u$  and  $v$  of size  $n$  is given at the left. [2✓]

$$p = \sum_{i=0}^{n-1} u_i v_i$$

4. Inheritance.

- Define a class **base** with an abstract virtual function called **value** that returns a **size\_t**. [1✓]
- Define a class **zero**, derived from **base**, which overrides that virtual function, so that it always returns the value 0. [1✓]
- Define a class **str**, derived from **base**, with a private string field and whose **value** function returns the **size** of the string. Do not show any members except those explicitly mentioned here. [2✓]

5. Complete the function `divide_by_2` as it would appear in `ubigint.cpp`. Follow the specifications for the programming project. A partial header file is shown at the left. [3✓]

```
class ubigint {
private:
    vector<char> data;
public:
    void divide_by_2();
};
```

6. Given the following in `intvec.h`, which defines a vector of integers. the field `data_` may point at a raw array of integers or it may be null.

```
class intvec {
private: int* data_;
        size_t size_;
public: intvec& operator= (const intvec&); // copier
        intvec& operator= (intvec&&); // mover
        ~intvec(); //destructor
};
```

- (a) Show the implementation of the *copy* `operator=` as it would appear in `intvec.cpp`. The data field of either object may or may not be null. [3✓]

- (b) Show the implementation of the *move* `operator=` as it would appear in `intvec.cpp`. The data field of the moved-from object is set to null. The data field of the moved-to object may or may not be null. [3✓]

- (c) Show the implementation of the destructor as it would appear in `intvec.cpp`. [1✓]

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		$\times 1 =$	$= a$
number of wrong answers		$\times \frac{1}{2} =$	$= b$
number of missing answers		$\times 0 =$	0
column total $c = \max(a - b, 0)$	12		$= c$

- What initializes `args` to the command line arguments, excluding the program name?  
`vector<string> args _____;`  
 (A) `(&argv[0], &argv[argc-1])`  
 (B) `(&argv[0], &argv[argc])`  
 (C) `(&argv[1], &argv[argc-1])`  
 (D) `(&argv[1], &argv[argc])`
- Given `string* s; string* t;` what is impossible?  
 (A) `s != t and *s != *t`  
 (B) `s != t and *s == *t`  
 (C) `s == t and *s != *t`  
 (D) `s == t and *s == *t`
- Function `f` does not modify its argument. Which declaration is most appropriate?  
 (A) `void f (const string&);`  
 (B) `void f (const string);`  
 (C) `void f (string&);`  
 (D) `void f (string);`
- What is the time efficiency of insertion into `std::map`?  
 (A)  $O(1)$   
 (B)  $O(\log n)$   
 (C)  $O(n)$   
 (D)  $O(n \log n)$
- As a class member, what makes the function `foo` abstract?  
 (A) `virtual void foo() = 0;`  
 (B) `virtual void foo() = abstract;`  
 (C) `virtual void foo() = default;`  
 (D) `virtual void foo() = explicit;`
- A \_\_\_\_\_ member of a class is visible to its own class and to any derived class, but not to other classes.  
 (A) `friend`  
 (B) `private`  
 (C) `protected`  
 (D) `public`
- Given the declarations `int* p; int i;` which expression is in error?  
 (A) `i - i`  
 (B) `i - p`  
 (C) `p - i`  
 (D) `p - p`
- What keyword is used to ensure compile-time computation of a value?  
 (A) `const`  
 (B) `constexpr`  
 (C) `final`  
 (D) `inline`
- What type is the result of dereferencing `map<string,int>::iterator`?  
 (A) `pair<const string, const int>`  
 (B) `pair<const string, int>`  
 (C) `pair<string, const int>`  
 (D) `pair<string, int>`
- The default `operator=` is probably inappropriate if a data member is a \_\_\_\_\_.  
 (A) class object  
 (B) complex number  
 (C) pointer  
 (D) primitive
- Which statement is syntactically correct, but has a narrowing conversion error?  
 (A) `char c (1000);`  
 (B) `char c <1000>;`  
 (C) `char c [1000];`  
 (D) `char c {1000};`
- What is the proper way to catch an exception called `exn`?  
 (A) `catch (exn e)`  
 (B) `catch (exn! e)`  
 (C) `catch (exn& e)`  
 (D) `catch (exn* e)`



The Antikythera mechanism, built ca. 150–100 BCE, is the oldest known complex scientific calculator, and is sometimes called the first known analog computer, with operational instructions written in Greek.  
[http://en.wikipedia.org/wiki/Antikythera\\_mechanism](http://en.wikipedia.org/wiki/Antikythera_mechanism)