CS100
Introduction to Programming

Lecture 30

C++ Examples

Organization of Final Exam

- Exam is closed-book and closed-notes
- Only a pen etc. is allowed
- The final exam will go over 50 points
- Material of entire semester is tested
- The time will be about 90 minutes
  (but it will be longer than the midterm exam,
  which was designed for 60 minutes)
- **Please remember your tutorial group number!!**

Organization of Final Exam

- Tentative!!

Multiple choice questions	Find the errors / bugs	C Problem-set	C++ Problem-set	Python Problem-set
12 points total: C: 4 points C++: 4 points Python: 4 points	14 points total: C: 4 points C++: 4 points Python: 6 points Each question is 2 points, 1 point for identifying the error/bug, and 1 point for a correct solution proposal	4 points total: Most likely just one question	10 points total: Note more than 2 questions	10 points total: Note more than 2 questions

Example Multiple Choice Questions

- Similar to questions in quizzes & mid-term exam

Question 3: What procedures of a parent class will be accessible in a child class?
(language: C++, 1 point)

[X] **public** procedures
[X] **protected** procedures
[ ] **private** procedures

Example Multiple Choice Questions

- Similar to questions in quizzes & mid-term exam

Question 4: Choose all of the following declarations that correctly overload a procedure with the declaration `float add( float input );` (language: C++, 1 point)

```
[ ]  float add( float newElement );
[ ]  double add( float newElement );
[X]  float add( float newElement1, float newElement2 );
[X]  float add( int newElement );
```

Example Multiple Choice Questions

- Example 3:
  - What are important ideas behind object-oriented programming?

    [x] Encapsulation
    [ ] Sequential program flows
    [x] Inheritance
    [ ] Concurrency

Example Multiple Choice Questions

- Example 4:
  - A function can be overridden if

    [ ] overriding happens in the same class.
    [x] overriding takes place in a child class.
    [ ] the function is not templated.
    [x] the overriding function has the same name.

Example Multiple Choice Questions

- Example 5:
  - A class with a purely virtual function

    [x] is called an abstract base class.
    [ ] is instantiatable.
    [x] needs to have that function implemented in a non-abstract derivate.
    [ ] can’t be inherited.

Example Multiple Choice Questions

- Example 6:
  - std::vector is preferred over a std::list if

    [ ] the number of elements is not known in advance.
    [x] random access is desired.
    [ ] frequent insertion of data happens.
    [x] a sorting problem has to be solved.

Example Error/Bug finding

- Example 1: Syntax errors
  - What are bugs in the following code? Correct them!

```
class Date {
public:
   Date_();
   int m_month;
   int m_day;
   int m_year;
};
```

Example Error/Bug finding

- Example 2: Bad memory management
  - What are the problems in the following code? Fix them

```
RndGenerator* getRndGenerator() {
  return new RndGenerator();
}

int main() {
  std::vector<int> numbers;
  for (int i = 0; i < 1000; ++i) {
    RndGenerator * p = getRndGenerator();
    numbers.push_back(p->nbr());
  }
  //do something with numbers
  return 0;
}
```

Missing size-reservation

Memory leak

循环里前后++

nbr()?

Example Error/Bug finding

- Example 3: Critical sections
  - What is the problem in the following code? Propose at least two solutions to this problem!

- Example 3:

```
#include <vector>
#include <thread>
#include <iostream>

class Counter {
public:
  Counter() {
    m_value = 0;
  };

  int getValue() {
    return m_value;
  };
  void increment() {
    ++m_value;
  };
private:
  int m_value;
};
```

Non-atomic operation

```
void incrementCounterManyTimes(
    Counter & counter ) {
  for( int i = 0; i < 5000; i++ ) {
    counter.increment();
  }
}

int main() {
  Counter counter;
  std::vector<std::thread> threads;
  for( int i = 0; i < 5; i++ ) {
    threads.push_back( std::thread(
        incrementCounterManyTimes,
        counter ) );
  }
  for( int i = 0; i < 5; i++ ) {
    threads[i].join();
  }
  std::cout << counter.getValue() << "\n";
  return 0;
}
```

Missing reference

Example Questions for C++ problem-set

- Example 1 (from quiz 2, Output prediction):

1. What is the output of the following code:

```
#include <iostream>
#include <vector>
#include <algorithm>

bool orderingFct( unsigned int & op1, unsigned int & op2) {
    if( (op1 % 2) != (op2 % 2) ) {
        if( (op1 % 2) ) return true;
        else return false;
    }
    else {
        if( op1 > op2 ) return true;
        else return false;
    }
}

int main() {
    std::vector<unsigned int> vec;
    for( unsigned int i = 1; i < 5; i++ )
        vec.push_back(i);
    std::sort( vec.begin(), vec.end(), orderingFct );
    for( int j; j < 4; j++ )
        std::cout << vec[j] << " ";
    return 0;
}
```

Need to be able to produce this by yourself, not just pick a choice!

(A) 4 3 2 1; (B) 3 1 4 2; (C) 2 4 1 3; (D) 1 3 2 4.

Example Questions for C++ problem-set

- Possible follow-up questions:
  - What if you would want the order to be … ? Please rewrite function … to meet this requirement.
  - What if you would want to use a list as a data container? Please rewrite the main function to use a list …

std::sort does not work with list,
need to use member function .sort()

Example Questions for C++ problem-set

- Example 2 (from quiz 2, Output prediction):

4. What is the output of the following code

```
#include <iostream>

class Integer {
public:
    Integer() {
        std::cout << "Integer ";
    };
    Integer( int i ) {};
};

class IntegerWrapper {
public:
    IntegerWrapper();
private:
    Integer m_i;
};
IntegerWrapper::IntegerWrapper() : m_i(0) {
    std::cout << "Wrapper ";
}

int main() {
    IntegerWrapper wrap;
}
```

Need to be able to produce this by yourself, not just pick a choice!

(A) Integer Wrapper; (B) Integer; (C) Wrapper Integer; (D) Wrapper.

Example Questions for C++ problem-set

- Possible follow-up questions:
  - What would happen if the constructor `Integer()` would not be defined? Would the program still work?

Example Questions for C++ problem-set

- Example 3 (code writing): Complete the following iterator implementation for **`std::vector`**

```
template<class T>
class VectorIterator : public std::iterator< std::random_access_iterator_tag, T > {
public:
  VectorIterator( std::vector<T> * vec = NULL, size_t index = 0 ) {
    m_vec = vec;
    m_index = index;
  };

  T & operator*() { … };
  T * operator->() { … };
  VectorIterator<T> & operator++() { … };
  VectorIterator<T> operator++(int) { … };
  VectorIterator<T> & operator--() { … };
  VectorIterator<T> operator--(int) { … };

  bool operator== ( const VectorIterator<T> & that ) const { … };
  bool operator!= ( const VectorIterator<T> & that ) const { … };
private:
  std::vector<T> * m_vec;
  size_t m_index;
};
```

See homework

Example Questions for C++ problem-set

- Example 4 (code writing): Write a template class which generalizes the following two functions!

```
void swap(char &x, char &y) {
    char t;
    t = x; x = y; y = t;
}
void swap(int &x, int &y) {
    int t;
    t = x; x = y; y = t;
}
void swap(float &x, float &y) {
    float t;
    t = x; x = y; y = t;
}
```

See slides