Polytechnic Tutoring Center

Final Exam Review - CS 2124, Spring 2025

Disclaimer: This mock exam is only for practice. It was made by tutors in the Polytechnic Tutoring Center and is not representative of the actual exam given by the CS Department. For more assistance with CS 2124, tutoring is available at the following times:

Monday	Tuesday	Wednesday	Thursday	Friday
10:00-8:00	10:00-2:00 4:00-6:00	10:00-8:00	10:00-2:00 4:00-8:00	10:00-5:00

Please answer all questions on the scratch papers provided. If you need extra pieces, just ask (loudly)!

1.
```
class A {
private:
string aString;
protected:
const string& get_string() const {
       return aString;
}
public:
A(const string& anotherString) : aString(anotherString) {}
};

class B : public A {
public:
B(const string& otherString) : A(otherString) {}
void display_string() const {
       cout << get_string() << endl;
}
};

class C : public A {
public:
C(const string& anotherotherString) : A(anotherotherString) {}
void display_string() const {
       cout << aString << endl;
}
};
int main() {
A a("A");
B b("B");
C c("C");
b.display_string();
c.display_string();
}
```

What is the result of the above program?

a. Runtime error as a result of Class B’s display_string method

b. Runtime error as a result of Class C’s display_string method

c. The output: B
C

d. Compilation error as a result of Class C’s display_string method

e. Compilation error as a result of Class B’s display_string method

2. What does the following code result in? And what key concept is used in this code?

```
class A {
private:
string aString;
public:
A(const string& anotherString) : aString(anotherString) {}
};

class B : public A {
public:
B(const string& otherString) : A(otherString) {}
};


int main() {
A* b = new B("B"); \\ line A
B* a = new A("A"); \\ line B
}
```

a. Results in a compilation error at line A; polymorphism

b. Results in a compilation error at line B; polymorphism

c. Results in a compilation error at line A; abstraction

d. Results in a compilation error at line B; abstraction

3. What will be the output of the following code?

```
class A {
private:
string aString;
protected:
const string& get_string() const {
       return aString;
}
public:
A(const string& anotherString) : aString(anotherString) {}
~A() { cout << "~A()\n"; }
};
```

```
class B : public A {
public:
B(const string& otherString) : A(otherString) {}
void display_string() const {
       cout << get_string() << endl;
}
~B() { cout << "~B()\n"; }
};


int main() {
A* a = new B("B");
B* b = new B("B2");
       delete a;
       delete b;
}
```

a. ~A()
~B()

b. ~A()
~B()
~A()

c. ~B()
~A()
~B()

d. ~B()
~A()
~B()
~A()

e. ~A()
~B()
~A()
~B()

---

4. What is the result of the following code? Output on the line below…

```
vector<int> vi = { 10, 2, 4, 1, 5, 3, 7, 9, 8, 6 };
```

```
list<int> li(3 + vi.begin(), vi.end());
for (int elem : li) {
cout << elem << " ";
}
```

––---

5. Write a templated function that enables you to find a specified value within any container with an iterator. You can assume the container only stores objects of one type. Note this “specified value”, along with the beginning and end should be passed into the function. If you couldn’t find the value, return a reasonable value.

––---

6. Write a recursive function that returns whether the string passed into the function is a palindrome. Note: you are allowed to use helper variables like the typical high and low frequently seen in data structures.

––---

7. Given the following tree node data structure implementation:

```
struct TNode {
   int data = 0;
   TNode* left = nullptr;
   TNode* right = nullptr;
};
```

Every TNode object lives on the heap. Write a recursive function, given a root node, free up the dynamically allocated memory by destroying all the nodes connected to it.

––---

8. Write an Iterator class that iterates over a singly linked list. Assume the iterator class you are writing is of the non-const type. Write the pre and post increment operators as well as the operator== and operator!= operators for the iterator. Assume the class has the following Node struct definition:

```
struct Node {
Node* next;
int data;
Node(Node* next = nullptr, int data = 0) : next(next), data(data) {}
};
```