BTLT-Individual-W08-02

a) Cho biết chương trình trên in gì ra màn hình? Giải thích trình tự gọi hàm trong hàm main().

Chương trình trên báo lỗi ở dòng code:

```
int main() {
    B obj1("text");
    A* obj2 = new B(obj1);
    foo(&obj1, *obj2);
}
```

Giải thích: Do không có constructor nào trong lớp B nhận đối số là const char*, mà ta đang
cố tạo một đối tượng B với một char*.

Sửa lỗi:

```
int main() {
    B obj1(strdup("text"));
    A* obj2 = new B(obj1);
    foo(&obj1, *obj2);
}
```

Sau khi sửa lỗi sẽ in ra màn hình console:

```
B: text
A: default
```

Trình tự gọi hàm trong hàm main():

- **B obj1(_strdup("text"));**
  Gọi đến constructor `B(char* s) : A(s) {};` gọi tiếp đến `A(char* s) { m_s = s; };`, nó sẽ khởi tạo m_s = “text”

- **A* obj2 = new B(obj1);**
  Code trên tạo một đối tượng của lớp A từ đối tượng obj1, nó sẽ gọi tới `B(const B& b) {};` để tạo ra một bản sao của obj1. Do không có đối số nào được truyền vào cho constructor cúa 'A' nên nó sẽ gọi đến `A() { m_s = _strdup("default"); };` và khởi tạo m_s là "default"

- **foo(&obj1, *obj2);**
  &obj1→display(): nó sẽ gọi tới display() của class A, prepare() của class B và in ra m_s là "text"
  Output: B: text
  ***obj2→display():** nó sẽ gọi tới display() của class A, prepare() của class A và in ra m_s là "default"
  Output: A: default

b) Hãy cho biết chương trình trên gặp vấn đề gì về bộ nhớ? Sửa lại cho đúng.
Chương trình trên gặp vấn đề về bộ nhớ vì khi truyền đối tượng B vào hàm foo, nó được truyền dưới dạng con trỏ A*. Khi hàm foo kết thúc, đối tượng obj2 được giải phóng bộ nhớ, nhưng không được giải phóng đúng do đó ta thêm vào lớp A một hàm hủy ảo.

Sửa lại: Trong class A ta thêm hàm hủy để giải phóng bộ nhớ

```
A() { m_s = _strdup("default"); };
A(char* s) { m_s = _strdup(s); };
virtual ~A() { delete[] m_s; }
```

c) Hãy trang bị operator >> cho lớp A để nhập chuỗi từ bàn phím cho thuộc tính m_s

```
friend std::istream& operator>>(std::istream& in, A& a) {
    char buffer[100];
    in.getline(buffer, 100);

    delete[]a.m_s;

    a.m_s = _strdup(buffer);
    return in;
}
```