

1. (10 points) Give the output for the following program:

```
1 #include <iostream>
2 class string {
3 public:
4     string()          { std::cout << "default" << std::endl;    }
5     string(const char*) { std::cout << "convert" << std::endl;    }
6     string(const string&) { std::cout << "copy" << std::endl;      }
7     ~string()          { std::cout << "destructor" << std::endl; }
8     string& operator=(const string&) {
9         std::cout << "assign" << std::endl;
10        return *this;
11    }
12 };
13 int main() {
14     string a("aarvark");
15     string* rope = new string("hang");
16 }
```

2. (10 points) Give the output for the following program:

```
1 #include <iostream>
2 void incrCount(int count) {
3     ++count;
4 }
5
6 void makeSwitch(int & count) {
7     switch ( count ) {
8         case 2: ++count;
9         case 3: ++count;
10        case 4: ++count;
11        case 5: ++count;
12        default: ++count;
13    }
14 }
15 int main() {
16     int count = 10;
17     incrCount(count);
18     std::cout << count << std::endl;
19
20     count = 8;
21     count = (count % 5) ? 2 : 3;
22     makeSwitch(count);
23     std::cout << count << std::endl;
24 }
```

3. (10 points) Give the output for the following program:

```
1 #include <iostream>
2 class string {
3 public:
4     string()          { std::cout << "default" << std::endl;    }
5     string(const char*) { std::cout << "convert" << std::endl;    }
6     string(const string&) { std::cout << "copy" << std::endl;      }
7     ~string()          { std::cout << "destructor" << std::endl; }
8     string& operator=(const string&) {
9         std::cout << "assign" << std::endl;
10        return *this;
11    }
12 };
13 int main() {
14     string x("cat"), y = x;
15 }
```

4. (10 points) The following program crashes with a double free error. Explain why.

```
1 #include <cstring>
2 #include <iostream>
3 class string {
4 public:
5     string(const char* s) : buf(new char[strlen(s)+1]) { strcpy(buf, s); }
6     ~string() { delete [] buf; }
7     const char* getBuf() const { return buf; }
8 private:
9     char * buf;
10 };
11
12 int main() {
13     string a("cat"), b = a;
14 }
```

5. (10 points) Give the output for the following program.

```
1 #include <iostream>
2 class Bird {
3 public:
4     Bird(int w) : wingSpan(w), speed(2*wingSpan) {
5         std::cout << "Speed: " << speed << std::endl;
6         std::cout << "Wing Span: " << wingSpan << std::endl;
7     }
8 private:
9     int speed;
10    int wingSpan;
11 };
12
13 int main() {
14     Bird robin(5);
15 }
```

6. (10 points) The following program compiles and executes but gives a warning. Give the output for the program. The warning given by g++ is:

```
main.cpp:3:21: warning: parameter x set but not used [-Wunused-but-set-parameter]
void initialize(int x, int y) {
                  ^
main.cpp:3:28: warning: parameter y set but not used [-Wunused-but-set-parameter]
void initialize(int x, int y) {
                  ^
```

```
1 #include <iostream>
2
3 void initialize(int x, int y) {
4     x = 7;
5     y = 8;
6 }
7
8 int main() {
9     int a = 10, b = 11;
10    initialize(a, b);
11    std::cout << a << std::endl;
12    std::cout << b << std::endl;
13 }
```

7. (40 points) The rule of three states that for a class that contains dynamically allocated data attributes (pointers), the programmer should write a destructor, a copy constructor, and an assignment operator. Write a class, `Student`, that contains a single data attribute: `char* name`. Write all three constructors and any additional functions needed to fulfill the rule of three. Also, overload an output operator for class `Student`, including any accessor functions needed (getters).