1. (25 points)

   (a) Give the output for the following program that compiles and executes without error or warning.

   (b) On line #16 the programmer is pushing an integer onto the vector vec. However, vec contains instances of Android (see line #14). How does line #16 work?

```
1  #include <iostream>
2  #include <vector>
3  class Android {
4  public:
5    Android() { std::cout << "default" << std::endl; }
6    Android(int) { std::cout << "convert" << std::endl; }
7    Android(const Android&) { std::cout << "copy" << std::endl; }
8    Android& operator=(const Android&) {
9      std::cout << "assign" << std::endl;
10     return *this;
11   }
12 };
13 int main() {
14   std::vector<Android> vec;
15   for (unsigned int i = 0; i < 2; ++i) {
16     vec.push_back( i );
17   }
18 }
```

convert
copy
convert
copy
copy
The conversion constructor is called on line #16.

2. (20 points)

   (a) Give the output for the following program.

   (b) What's the biggest improvement that you could make to line #14.

```
1  #include <iostream>
2  class Android {
3  public:
4    Android() { std::cout << "default" << std::endl; }
5    Android(int) { std::cout << "convert" << std::endl; }
6    Android(const Android&) { std::cout << "copy" << std::endl; }
7    Android& operator=(const Android&) {
8      std::cout << "assign" << std::endl;
9      return *this;
10   }
11 };
12 class CyberLife {
13 public:
14   CyberLife(const Android& p) { mon = p; }
15 private:
16   Android mon;
17 };
18 int main() {
19   CyberLife connor(800);
20 }
```

```
convert
default
assign
Use initialization list
```

---

3. (5 points) Give the output for the following program.

```
1  #include <iostream>
2  #include <cstring>
3
4  class string {
5  public:
6    string(const char * s) : buf(new char[strlen(s)+1]) {
7      strcpy(buf, s);
8    }
9    char* getBuf() const { return buf; }
10 private:
11   char *buf;
12 };
13
14 int main() {
15   string x("cat");
16   char* buf = x.getBuf();
17   buf[0] = 'r';
18   std::cout << x.getBuf() << std::endl;
19 }
```

```
rat
```

4. (20 points) Give output for the following program that compiles and executes without error or warning.

```
#include <iostream>
#include <cstring>
class string {
public:
  string() { std::cout << "default" << std::endl; }
  string(const char*) { std::cout << "convert" << std::endl; }
  string(const string&) { std::cout << "copy" << std::endl; }
  ~string() { std::cout << "destructor" << std::endl; }

  string& operator=(const string&) {
    std::cout << "copy assign" << std::endl;
    return *this;
  }
};

void fun(string) { }

int main() {
  string cat("cat"), dog = cat;
  fun(cat);
  int x = 17;
  int& r = x;
  r = 99;
  std::cout << x << std::endl;
}
```

convert
copy
copy
destructor
99
destructor
destructor

---

5. (10 points) Write a copy assignment operator for class Pokemon.

```
#include <iostream>

class Pokemon {
public:


private:
  char* name;
};
```

```
Pokemon& operator=(const Pokemon& rhs) {
  if ( this == &rhs ) return *this;
  delete [] name;
```

```
  name = new char[strlen(rhs.name)+1];
  strcpy(name, rhs.name);
  return *this;
}
```

6. (20 points)

- What is grammarware?
  Any software that uses a grammar or is comprised of a grammar

- Basically, what's the "research agenda" described in the paper?
  To leverage engineering principles to develop, modify, test,
  and maintain grammars and grammarware

- The authors claim that "Grammarware seems to be second-class software." What do they mean by this? (for example, consider refactoring)
  There's a plethora of research to develop techniques to comprehend, modify, extend, test, and maintain software. However, there is very little research that attempts to apply these principles to grammars and grammarware. Thus, grammars and grammarware are second class citizens to software in general.

- The authors ask the question: "what is a good grammar?" How would you address this issue?
  I would investigate software metrics and attempt to apply them to grammars. I would do this in an effort to judge the quality of the grammar under study.

- Write a regular expression for the following right regular grammar:

A : a A | B
B : b B | λ
a^*b^*

- Write a regular expression for a valid identifier in C^{++}
  $[a-zA-Z][a-bA-Z0-9\_]$