Carleton University School of Computer Science
COMP 2404 Fall 2017 Midterm Test -ANSWERS

(Scantron Test: answers to be submitted on the Scantron card provided)
All questions worth 1 mark (15 marks total).

Part I. Questions about Software Development and Software Engineering

1. Which ONE of the following statements about software requirements is **true**?

(A) Unambiguous requirements can be obtained by asking the customer to provide them.
(B) Requirements written in English pertain to the design models but not to the code.
(C) Developers usually can't agree on what a requirement is (i.e. what the definition "requirement" is) .
(D) Development processes usually assume the requirements are fully debugged before coding starts.
(E) Itemized and categorized requirements are artifacts common to most software development processes.

2. Which ONE of the following statements is **true** about requirements categorized as "Functional Requirements"?

(A) They pertain to legal and demographic conditions in which the app must operate.
(B) They pertain to behavior the users or client would perceive.
(C) They pertain to coding constraints the developer must adhere to.
(D) They pertain to what functions, or instance methods, the code must provide.
(E) They pertain to features typically put in `private` sections of C++ code.

3. Which ONE of the following statements is **true** about the Waterfall development process model?

(A) Requirements debugging must be finished before construction coding begins.
(B) Customers are made part of the development team to help clarify requirements.
(C) It uses itemized requirements so that it's easier to make changes to the requirements once construction coding has begun.
(D) It's based on the metaphor that water sometimes flows upstream in turbulent conditions.
(E) It is the recommended process to use when customers don't know exactly what their requirements are.

4. Which ONE of the following statements is **false** about the core concepts of the Agile development process model?

(A) Values individuals and interactions over formal process and tools.
(B) Values comprehensive documentation over code that appears to work.
(C) Values customer collaboration over contract negotiation.
(D) Values responding to change over following a prescribed plan.
(E) Values many small iterations over few large ones.

5. Which ONE of the following statements is **false** about the how software quality is typically defined in Software Development/Software Engineering?

(A) Quality Software is software that satisfies specified and itemized requirements.
(B) Quality is "Fitness for Purpose".
(C) Software quality accounts for the needs and expectations of the customer or client.
(D) The definition of software quality applies to more than one process model (e.g. Waterfall, Agile, Spiral etc.).
(E) Achieving quality software is a primary goal of most development process models.

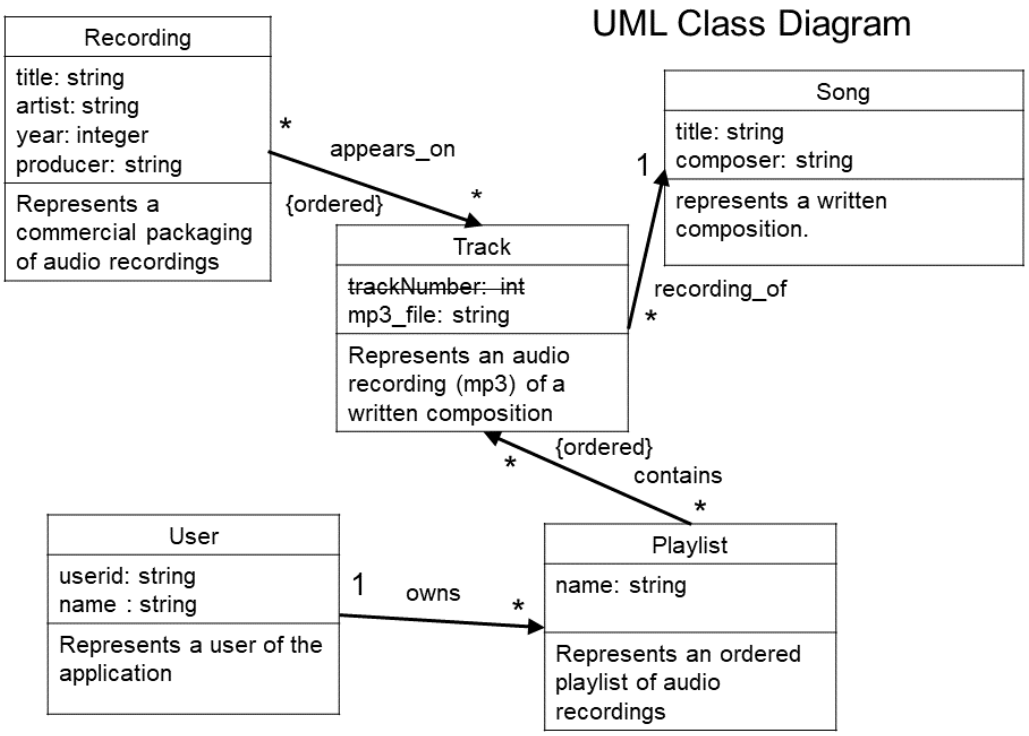
6. Object-oriented development processes are often described in terms of the following activities: Requirements Capture, Analysis, Design, Implementation, and Testing. Which ONE of the following statements is **false** about these activities?

(A) In Requirements Captures the customer's requirements are understood as a set of interacting objects.
(B) In Analysis a model is created that could solve the customers problem using interacting objects.
(C) In Design a model is created that consists of objects that could be built in the intended programming language and under the system constraints.
(D) In Implementation code is written that implements the objects specified in the design.
(E) In Testing the code is validated and verified against the specified requirements.

7. Which ONE of the following statements is **false** about what Software Engineering is or attempts of achieve?

(A) Software engineering hopes to achieve safe and predictable development by applying engineering techniques to software development.
(B) Software engineering uses both natural language requirements and pictorial models to describe problems.
(C) Software engineering uses the same "Engineering Method" as is used in other types of professional engineering disciplines like civil engineering, aeronautical engineering.
(D) The term "Software Engineering" is controversial because it is not clear that it is a true engineering discipline.
(E) There are restrictions in various states and provinces on who is allowed to call themselves a "Software Engineer".

8. Consider the following UML structure model similar to the one from assignment #2.



Which ONE of the following statements is **false** about the correct interpretation of this model?

(A) The square boxes represent classes not objects.
(B) A Track object can be an audio recording of at most one song.
(C) Several different Track objects can be recorded versions of the same Song.
(D) The deletion of a Recording object will create a referential integrity error.
(E) Playlists need not contain any Track references.

Part II. C++ Basics

9. Given the class definition below, which ONE of the following would be the best (most correct) signature for **`OPERATOR<<SIGNATURE`**?

```
#include <iostream>
#include <string>
using namespace std;

class Person {
  string theirname;
public:
Person(string aName) : theirname(aName) {}
string getName() const {return theirname;}
void printOn(ostream & out) const {out << theirname;}
};
OPERATOR<<SIGNATURE {
   x.printOn(o);
   return o;
}

int main(){
  Person p1("Lou"), p2("Sue");
  cout << p1 << "\n" << p2 << endl;  return 0;
}
```

(A) `ostream & operator<<(ostream & o, Person & x);`
(B) `ostream & operator<<(const ostream & o, Person & x);`
(C) `ostream & operator<<(ostream & o, const Person & x);`
(D) `ostream & operator<<(const ostream & o, const Person & x);`
(E) `ostream operator<<(const ostream & o, const Person & x);`

10. Consider a class `Set` with `private` variables `int size` and `vector<BankAccount *> buffer` that models a collection of `BankAccount` objects. It is required that the same object, not a copy, can be accessed and so the set stores pointers to the `BankAccount` objects. Suppose an `addElement()` instance method of `Set` is used to insert objects. Which ONE of the following would be the correct signature for the `addElement()` method to be consistent with the code below?

```
BankAccount b1, b2, b3;
Set s;
s.addElement(b1).addElement(b2).addElement(b3);
s.getFirst().deposit(100);
```

```
A) Set & addElement(BankAccount & item);
B) const Set & addElement(BankAccount &  item);
C) Set & addElement(const BankAccount &  item) const;
D) void addElement(BankAccount &  item);
E) void addElement(BankAccount *  item);
```

The next two questions pertain to the code example below that was compiled and run with the following command `g++ -g -Wall main.cpp`.

```
#include <iostream>
using namespace std;

class X {
private:
  int n;
public:
  X(void) : n(0) {cout << "X(void) ";}
  X(int i) : n(i) {cout << "X(int) ";}
  X(const X& a) : n(a.n){cout << "X(const X&) "; }
  ~X(void) {cout << "~X() ";}
};

class Y {
private:
  int n;
  X anX;
public:
  Y(void) : n(0) {cout << "Y(void) ";}
  Y(int i): n(i), anX(i)  {cout << "Y(int) ";}
  Y(const X & x) :  n(0), anX(x)  {cout << "Y(X&) ";}
  Y(const Y & a) : n(a.n),anX(a.anX){cout << "Y(const Y&) "; }
  ~Y(void) {cout << "~Y() ";}
};

int main() {
   X x1(42); cout << "\n";    //Line 1
   Y y1;     cout << "\n";    //Line 2
   Y y2=y1;  cout << "\n";    //Line 3
   Y y3(x1); cout << "\n";    //Line 4
   Return 0;
}
```

11. Which ONE of the following is the output from the statement labeled: `//Line 2`?

A) X(void) Y(void)
B) X(const X&) Y(void)
C) X(const X&) Y(const X&)
D) X(const X&) Y(X&)
E) Y(X&) X(const X&)

12. Which ONE of the following is the output from the statement labeled: `//Line 3`?

A) X(Y&)
B) Y(const Y&)
C) X(const X&)
D) Y(void)
E) X(const X&) Y(const Y&)

Part III C++ Programming

The following is a program that represents `Date`, `EmailAddress`, `Person` and `Collection` classes. `Date` and `EmailAddress` objects store their private variables **by value**, but the `Person` and `Collection` classes use **heap** storage (allocated with `new`). The program compiled with the g++ compiler and ran, but is shown with some code missing from the `Person` class (all other classes are complete). The output of the program is shown after the code. Study this code and its output and then answer the questions that ask you to choose the correct implementation of the **`//MISSING CODE`** in the `Person` class. Your solutions must be free of memory problems. (That is, don't reference heap objects that have been deleted and don't leave memory objects on the heap or do double deletions etc.)

```
#include <iostream>
#include <string>
using namespace std;

class EmailAddress{
public:
EmailAddress( const string & aUserID, const string & aDomain) {
   userID = string(aUserID);
   domain = string(aDomain);
}
EmailAddress( const EmailAddress & anEmailAddress) {
   userID = string(anEmailAddress.userID);
   domain = string(anEmailAddress.domain);
}
EmailAddress & operator=(const EmailAddress & anEmailAddress) {
   if(&anEmailAddress != this) {
      userID = string(anEmailAddress.userID);
      domain = string(anEmailAddress.domain);
   }
   return *this;
}
~EmailAddress(){
   //do nothing
}
void printOn(ostream & ostr) const {ostr << userID << "@" << domain;
}

private:
     string userID;
     string domain;
};

ostream & operator<<(ostream & ostr, const EmailAddress &
anEmailAddress) {
     anEmailAddress.printOn(ostr);  return ostr;
}
```

```
class Date{
public:
   Date(const string & aDateString) : dateStr(aDateString) {}
   Date(const Date & aDate) :  dateStr(aDate.dateStr) {};
   void printOn(ostream & out) const {out << dateStr;}
private:
   string dateStr; //represents date as a literal string
};
ostream & operator<<(ostream & ostr, const Date & d) {
   d.printOn(ostr);
   return ostr;
}


class Person {
public:
Person(const string & aName, const string & userID, const string &
domain, Date aDate) {
   //create heap objects
   name = new string(aName);
   emailAddress = new EmailAddress(userID, domain);
   birthday = new Date(aDate);
}

Person(const Person & p) {
  //MISSING CODE --write the copy constructor
}

~Person() {
  // MISSING CODE --write the destructor
}

Person & operator=(const Person & p) {
  //MISSING CODE --write the assignment operator.
}

void printOn(ostream & ostr) const {
  ostr << *name << " (" << *birthday << ") " << *emailAddress;
}

private:
   string * name; //pointer to a person's name
   EmailAddress * emailAddress; //pointer to a person's email address
   Date * birthday; //pointer to person's birthday

}; //end class Person

ostream & operator<<(ostream & out, const Person & p) {
   p.printOn(out);
   return out;
}
```

```
//class Collection
const int INITIAL_COLLECTION_SIZE = 12;

typedef Person T; //T is an alias for Person
class  Collection {
public:
Collection (int len = INITIAL_COLLECTION_SIZE) { //constructor
       cout << "Collection(int)\n";
      mysize = len;
      buffer = new T*[len];
       for(int i=0; i<mysize; i++) buffer[i] = NULL;
}
Collection (const Collection &c) {
   mysize = c.mysize;
   buffer = new T*[mysize];
   for(int i=0; i<mysize; i++)
     buffer[i] = c.buffer[i];
}
Collection & operator=(const Collection & v) {
   if(&v != this) {
     delete [] buffer;
     mysize = v.mysize;
     buffer = new T*[mysize];
     for(int i=0; i<mysize; i++)
        buffer[i] = v.buffer[i];
      }
     return *this;
}
~Collection (void) { delete  [] buffer; }

int capacity (void) { return mysize; }

Collection & addElementAt(T & x, int index) {
   if(index < mysize) buffer[index] = &x;
   return *this;
}
T& elementAt(int index) { return *buffer[index] ;}

void printOn(ostream & o) const {
//This method should print all the non NULL elements
   o << "Contents:\n";
   for (int i=0; i<mysize; i++)
      if(buffer[i] != NULL)  o << *buffer[i] << "\n";
}
private:
   int mysize;
   T **buffer;
}; //end class Collection

ostream & operator<<(ostream & o, const Collection & c) {
    c.printOn(o); return o;
}
```

```
//The main program and its output

int main( ) {
Person alan("Alan Day", "alan", "scs.carleton.ca", Date("2 July
1961"));
Person sue("Sue Smith", "Sue", "hotmail.com", Date("30 June 1974"));
Person lou("Lou Nel", "ldnel", "scs.carleton.ca", Date("29 July
1977"));

Collection people;
people.addElementAt(alan,0).addElementAt(sue,1).addElementAt(lou,2);
cout << people;

//test initialization and assignment of Collection class
Collection  morePeople = people;

Collection  others;
others = people;
cout << others;
return 0;
}
```

```
//OUTPUT
/*
Collection(int)
Contents:
Alan Day (2 July 1961) alan@scs.carleton.ca
Sue Smith (30 June 1974) Sue@hotmail.com
Lou Nel (29 July 1977) ldnel@scs.carleton.ca
Collection(int)
Contents:
Alan Day (2 July 1961) alan@scs.carleton.ca
Sue Smith (30 June 1974) Sue@hotmail.com
Lou Nel (29 July 1977) ldnel@scs.carleton.ca
*/
```

13. Which ONE of the following is the correct implementation of the `Person` class copy constructor?

```
Person(const Person & p) {

  //OPTION A
  name = p.name;
  birthday = p.birthday;
  emailAddress = p.emailAddress;

  //OPTION B
  name = new p.name;
  birthday = new p.birthday;
  emailAddress = new p.emailAddress;

  //OPTION C
  name = new string(p.name);
  birthday = new Date(p.birthday);
  emailAddress = new EmailAddress(p.emailAddress);

  //OPTION D
  name = new string(*p.name);
  birthday = new Date(*p.birthday);
  emailAddress = new EmailAddress(*p.emailAddress);

  //OPTION E
  name = new char[strlen(p.name) + 1];
  strcpy (name, p.name);
  birthday = new Date(strlen (p.birthday + 1));
  strcpy (birthday, p.birthday);
  emailAddress = new EmailAddress(strlen(p.emailAddress + 1));
  strcpy (emailAddress, p. emailAddress);

}
```

14. Which ONE of the following is the correct implementation of the `Person` class destructor?

```
~Person() {

// OPTION A
    delete birthday;
    delete emailAddress;
    delete name;

// OPTION B
    delete [] birthday;
    delete [] emailAddress;
    delete [] name;

// OPTION C
    delete name;

// OPTION D
    delete &birthday;
    delete &emailAddress;
    delete &name;

// OPTION E
    // do nothing

}
```

15. Which ONE of the following is the correct implementation of the `Person` class assignment operator?

```
Person & operator=(const Person & p) {

//OPTION A
      birthday = p.birthday;
      name = p.name;
      emailAddress = p.emailAddress;
      return *this;

//OPTION B
      birthday = new Date(*p.birthday);
      name = new string(*p.name);
      emailAddress = new EmailAddress(*p.emailAddress);

//OPTION C
   if(&p != this) {
      birthday = new Date(*p.birthday);
      name = new string(*p.name);
      emailAddress = new EmailAddress(*p.emailAddress);
   }
   return *this;

//OPTION D
   if(&p != this) {
      delete birthday;
      delete name;
      delete emailAddress;
      birthday = new Date(*p.birthday);
      name = new string(*p.name);
      emailAddress = new EmailAddress(*p.emailAddress);
      }
   return *this;

//OPTION E
      delete birthday;
      delete [] name;
      delete emailAddress;
      birthday = new Date(*p.birthday);
      name = new char[strlen(p.name) + 1];
      strcpy (name, p.name);
      emailAddress = new EmailAddress(*p.emailAddress);
      return *this;

}
```