COMP 2404

Midterm Exam

18/50 => 36%

[out of 50 marks]

Duration: 80 minutes

Authorized Memoranda: NONE

Name: [redacted]

[redacted]

Multiple Choice Questions (circle ONE answer for each question) **[10 marks]**

1. Given the program in Figure 1, which of lines (23) to (25) are valid and will not result in an error:

   a. (23) and (24) X
   b. (25)
   c. (24)
   d. (23) X

   2

2. Given the program in Figure 1, which of lines (27) to (30) are valid and will not result in an error:

   a. (27) and (28)
   b. (27)
   c. (29) and (30)
   d. (30)

   b X

3. Given the program in Figure 1, which of lines (32) to (34) are valid and will not result in an error:

   a. (32)
   b. (32) and (34)
   c. (33)
   d. (34)

   a X

4. Given the program in Figure 1, which of lines (36) to (38) are valid and will not result in an error:

   a. (36) and (37)
   b. (38)
   c. (37)
   d. none

   b X

5. Given the program in Figure 1, which of lines (40) to (42) are valid and will not result in an error:

   a. all of them
   b. none
   c. (40) and (41)
   d. (41)

   d X

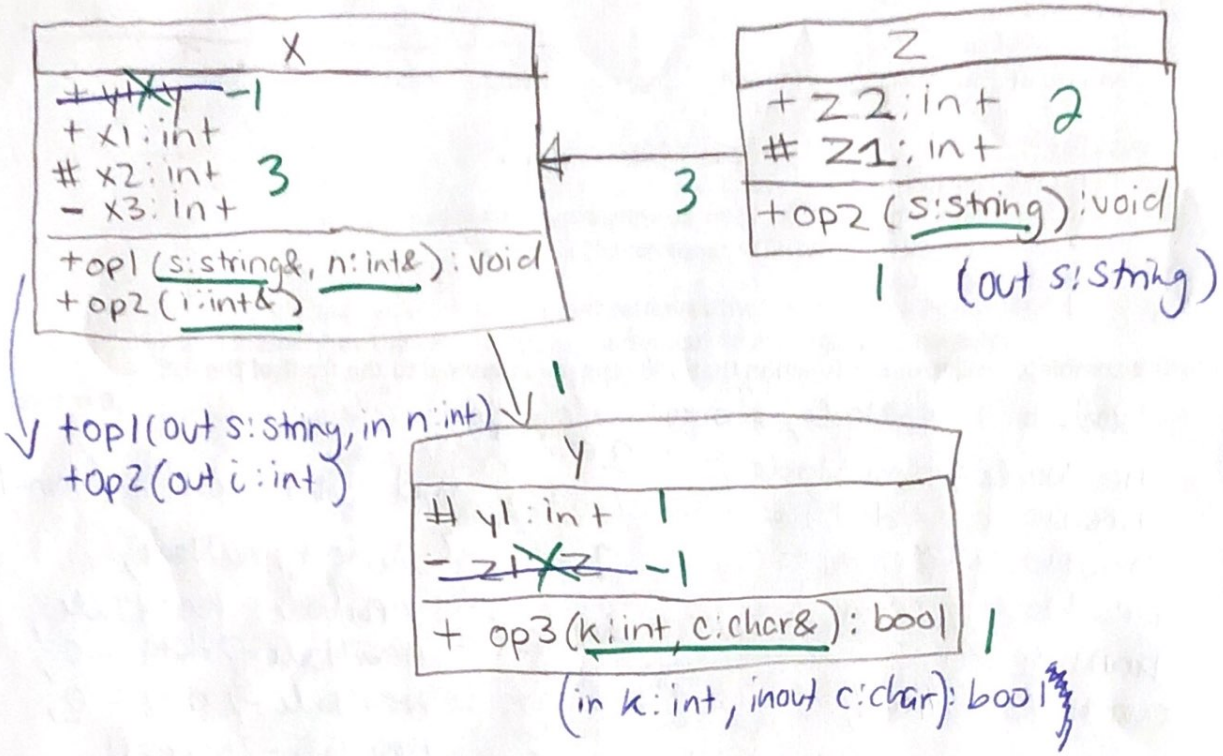
2

```
01  class Y {
02    public:    bool op3(int k, char& c) { if (k < 10 && c > 0) c = 'J'; }
03    protected: int y1;
04    private:   Z  z1;  };
05
06  class X {
07    public:
08      void op1(string& s, int& n) { if (n > 0) s = "abc"; }
09      void op2(int& i) { i = 99; }
10      Y    y1;  int x1;
11    protected:  int x2;
12    private:    int x3;  };
13
14  class Z : public X {
15    public:     void op2(string s) { s = "xyz"; }
16                int  z2;
17    protected:  int z1;  };
18
19  int main() {
20    X x;  Y y;  Z z;
21    string tmp = "hello"; int n1 = 12; int n2 = 34; char c  = 'C';
22
23    z.op2(n1);
24    z.op2(tmp);
25    x.op2(tmp);
26
27    z.x1 = 0;
28    z.x2 = 0;
29    x.z1 = 0;
30    x.z2 = 0;
31
32    x.y1.op3(n2,c);
33    y.x1.op2(n1);
34    y.z1.op2(tmp);
35
36    y.z1.op2(n1);
37    y.z1.op2(tmp);
38    z.y1.op3(n2,c);
39
40    z.y1.y1 = 0;
41    z.y1.op3(n2,c);
42    y.z1.z1 = 0;
43
44    return 0;
45  }
```

Figure 1

UML Question **[20 marks]**

6. Draw the UML diagram corresponding to the classes defined in Figure 1. You must show all classes, and all attributes, operations, associations, directionality, and multiplicity, where applicable:



10

10/9

Programming Questions [20 marks]

7. Given the class definition below:

```
class Dlist  {
  class Node {
    friend Dlist;
    private:  Animal* data;  Node*   prev;  Node*   next;
  };
  public:
    Dlist();  ~Dlist();
    void pushBack(Animal*);  void pushFront(Animal*);
    Animal* popBack();  Animal* popFront();
  private:  Node* head;
};
```

write the complete `pushFront()` function that adds the given animal to the front of the list. [8 marks]

```
Node = *newNode, * prevNode, *currNode;
newNode = new Node,
newNode -> data = animal;
newNode -> prev = 0;
newNode -> curr = 0;
prevNode = 0;
currNode = head
(while currNode != 0) {
    prevNode = currNode;
    currNode = currNode -> next;
}
if (prevNode = 0) {
  head = newNode;
} else {
  prevNode = newNode;
}
newNode -> next = currNode;
```

```
void Dlist:: pushFront(Animal *c)
{ Node * newNode;
  newNode = new Node;
  newNode -> data = c;
  newNode -> prev = 0;
  newNode -> next = 0;
  newNode -> next = head
  if (head != 0)
    head -> prev = newNode;
  head = newNode
}
```

4

4

marks]

8. Given the class definition below:

```
class Tlist  {
  class Node {
    friend Tlist;
    private:  Animal* data;  Node* next;
  };
  public:
    Tlist();  ~Tlist();
    void pushBack(Animal*);  void pushFront(Animal*);
    Animal* popBack();  Animal* popFront();
  private:  Node* head;  Node* tail;
};
```

write the complete `popBack()` function that removes the last element at the back of the list and returns that element as the return value. Remember to manage your memory.

[12 marks]

```
Animal * removedAnimal;
IF (currNode == 0) {
  return 0;
} else {
  tail = removedAnimal;
  tail = prevNode;
}
return removedAnimal;
```

X

```
{ Animal *goner;
Node * currNode;
Node * prevNode;
if (head == 0)
    return 0;
currNode = head;
prevNode = 0;
while (currNode != tail) {
  prevNode = currNode;
  currNode = currNode->next;
}
goner = tail->data;
delete tail;
if (prevNode == 0) {
  head = tail = 0;
} else { prevNode->next = 0;
         tail = prevNode;
}
return goner;
}
```

2