Midterm Exam

Started: Jun 3 at 3:47pm

Quiz Instructions

Answer the following questions. The questions are of the following types:

- Matching

- Multiple choices (with only one correct answer)

- Multiple answers (with potentially more than one correct answer)

Question 1

4 pts

The following code gives the definition of class **DogLicense**.

Match the numbered blank spaces with the given options.

```
---1---  ---2--- {
---3---:
        void SetYear(int yearRegistered);
        void CreateLicenseNum(int customID);
        int GetLicenseNum() const;
---4---:
        int licenseYear;
        int licenseNum;
};
```

---1--- [Choose]

---2--- [Choose]

---3--- [Choose]

---4---

[Choose]

Question 2

2 pts

Consider the class definition from the previous question.

What would be the non-inline definition of function **GetLicenseNum()** ?

- ○ **int GetLicenseNum() const {**
  return licenseNum;
  }
- ○ **int DogLicense::GetLicenseNum() {**
  return licenseNum;
  }
- ○ **int GetLicenseNum() {**
  return licenseNum;
  }
- ○ **int DogLicense::GetLicenseNum() const {**
  return licenseNum;
  }

Question 3

2 pts

Consider the class definition **DogLicense** in Question 1.

Assume that a new **string** data member **DogBreed** has been added to this class.

What would be the appropriate declaration of the setter **SetDogBreed**?

- ○ string SetDogBreed(string dogBreed);
- ○ void SetDogBreed(string dogBreed);
- ○ void SetDogBreed(string dogBreed) const;
- ○ string SetDogBreed();

Question 4

2 pts

Consider the class definition **DogLicense** in previous question.

Select the answer that would provide inline default constructor and the constructor with arguments for all the data members.

○
```
void DogLicense() {
        licenseYear = 0;
        licenseNum = 0;
        dogBreed = "N/A";
    }
    void DogLicense(int licenseYear, int licenseNum, string dogBreed) {
        licenseYear = licenseYear;
       licenseNum = licenseNum;
      dogBreed = dogBreed;
    }
```

○
```
DogLicense() {
        licenseYear = 0;
        licenseNum = 0;
        dogBreed = "N/A";
    }
    DogLicense(int licenseYear, int licenseNum, string dogBreed) {
        this->licenseYear = licenseYear;
        this->licenseNum = licenseNum;
        this->dogBreed = dogBreed;
    }
```

○
```
DogLicense() const {
        licenseYear = 0;
        licenseNum = 0;
        dogBreed = "N/A";
    }
    DogLicense(int licenseYear, int licenseNum, string dogBreed) const {
        this->licenseYear = licenseYear;
        this->licenseNum = licenseNum;
        this->dogBreed = dogBreed;
    }
```

○
```
void DogLicense() {
        licenseYear = 0;
        licenseNum = 0;
        dogBreed = "N/A";
    }
    void DogLicense(int licenseYear, int licenseNum, string dogBreed) {
        this->licenseYear = licenseYear;
```

```
        this->licenseNum = licenseNum;
        this->dogBreed = dogBreed;
    }
```

Question 5

4 pts

Consider the class definition **DogLicense** in previous question.

Class **LimitedDogLicense** is a **DogLicense** with expiration year.

Select all the appropriate definitions of this class?

- [ ]
```
class public LimitedDogLicense : DogLicense {
public:
        ...
protected:
        int expirationYear;
};
```

- [ ]
```
class LimitedDogLicense : public DogLicense {
public:
        ...
private:
        int expirationYear;
};
```

- [ ]
```
class LimitedDogLicense : protected DogLicense {
public:
        ...
private:
        int expirationYear;
};
```

- [ ]
```
class public LimitedDogLicense : DogLicense {
public:
        ...
private:
        int expirationYear;
};
```

Question 6

2 pts

Consider the class definition **LimitedDogLicense** in previous question.

Suppose the expiration year in a **LimitedDogLicense** is two years from the registration year.

What would be the best inline definition of **SetYear** method in this class?

- ◯ **void override SetYear(int yearRegistered) {**
  DogLicense::SetYear(yearRegistered);
  this->expirationYear = yearRegistered + 2;
  }
- ◯ **void SetYear(int yearRegistered) override {**
  DogLicense::SetYear(yearRegistered);
  this->expirationYear = yearRegistered + 2;
  }
- ◯ **void SetYear(int yearRegistered) {**
  this -> licenseYear= yearRegistered ;
  this->expirationYear = yearRegistered + 2;
  }
- ◯ **void SetYear(int yearRegistered) override {**
  this -> licenseYear= yearRegistered ;
  this->expirationYear = yearRegistered + 2;
  }

Question 7

1 pts

Consider the class definition **DogLicense** in previous question.

Which one of the following statements is syntactically correct?

- ◯ **DogLicense& dl = new DogLicense();**
- ◯ **DogLicense dl = new DogLicense();**
- ◯ **DogLicense ^dl = new DogLicense();**
- ◯ **DogLicense* dl = new DogLicense();**

Question 8

1 pts

Consider the previous question.

In which memory section, the space for **DogLicense** object has been allocated?

- ◯ Stack
- ◯ Heap
- ◯ Code
- ◯ Static

Question 9

2 pts

Consider the previous question.

Select all the correct syntax to set the registration year of the object associated with **dl** to 2020.

- ☐ **dl = SetYear(2020);**
- ☐ **dl.SetYear(2020);**
- ☐ **dl->SetYear(2020);**
- ☐ **(*dl).SetYear(2020);**

Question 10

2 pts

Consider the following code.

Select all the possible outputs.

```
int numArray[] = { 1, 2, 3, 4, 5, 6 };
for (int i = 1; i <= 6; i++) {
      cout << numArray[i] << " ";
}
```

- [ ] **2 3 4 5 6 -858993460**
- [ ] **2 3 4 5 6 0**
- [ ] **Run time error**
- [ ] **1 2 3 4 5 6**

Question 11

2 pts

Consider the following vector definition.

Which one of these will swap elements in indexes **i** and **j**?

vector<DogLicense> v(8);

- ○ **DogLicense temp = v.at(j);**
  v.at(j) = temp;
  v.at(j) = v.at(i);
- ○ **DogLicense temp = v.at(i);**
  v.at(j) = temp;
  v.at(i) = v.at(j);
- ○ **DogLicense temp = v.at(i);**
  v.at(i) = v.at(j);
  v.at(j) = temp;
- ○ **DogLicense temp = v.at(j);**
  v.at(i) = temp;
  v.at(j) = v.at(i);

Question 12

2 pts

Which one of the following is a function that compares two values of same type and determines if the first value is smaller than the second one?

○ **template<typename T>**
bool smaller(T v1, T v2) {
return v1 < v2;
}

○ **typename <template T, template Y>**
bool smaller(T v1, Y v2) {
return v1 < v2;
}

○ **template<typename T, typename Y>**
bool smaller(T v1, Y v2) {
return v1 < v2;
}

○ **typename <template T>**
bool smaller(T v1, T v2) {
return v1 < v2;
}

Question 13

2 pts

What would be the order of the elements of the list after applying these lines of code?

```
list<int> l = { 5, 8, 3, 2 };
l.pop_back();
l.push_front(2);
l.push_back(7);
l.front();
```

○ 7, 5, 8, 3

○ 2, 5, 8, 3, 7

○ 7, 5, 8, 3, 2

○ 5, 8, 3, 7

Question 14

2 pts

Suppose we want to define a pair **p** with **true** as its first component and **5** as its second.

Select all the correct answers.

- [ ] **pair<bool, int> p;**
  p.first(true);
  p.second(5);
- [ ] **pair<bool, int> p;**
  p = make_pair(true, 5);
- [ ] **pair<bool, int> p;**
  p.make_pair(true, 5);
- [ ] **pair<bool, int> p;**
  p.first = true;
  p.second = 5;

Question 15

2 pts

What would be the elements in the map **m** after the following lines of code.

```
map<int, string> m;
m.emplace(3, "Bill");
m.emplace(6, "Ted");
m.emplace(3, "Andy");
m.emplace(7, "Emma");
m.at(7) = "Jack";
m.erase(4);
```

- ○ (3, "Bill")
  (6, "Ted")
  (7, "Jack")
- ○ (3, "Andy")
  (6, "Ted")
  (7, "Jack")
- ○ Error
- ○ (3, "Bill")

(6, "Ted")

(7, "Emma")

Question 16

2 pts

Consider the following set **s**.

set<int> s = { 4, 5, 2, 6 };

What is the return value of **s.insert(5)** ?

- ◯ pair of iterator at 5, and true
- ◯ true
- ◯ pair of iterator at 5, and false
- ◯ false

Question 17

2 pts

Consider a queue of integers **q**, defined as follow.

queue<int> q;

Which one of the following would print the elements of **q** without changing it?

- ◯
```
for (i = 0; i < q.size(); i++) {
        cout << q.front();
        q.push(q.front());
        q.pop();
}
```
- ◯
```
for (i = 0; i < q.size(); i++) {
        cout << q.back();
        q.pop();
        q.push(q.back());
}
```
- ◯
```
for (i = 0; i < q.size(); i++) {
        cout << q.back();
        q.push(q.back());
```

```
    q.pop();
}
```

○
```
for (i = 0; i < q.size(); i++) {
    cout << q.front();
    q.pop();
    q.push(q.front());
}
```

Question 18

2 pts

Consider a stack of integers **stack**, defined as follow.

deque<int> stack;

Which one of the following would print the elements of **stack** without changing it?

☐
```
deque<int> tempStack;

for (int i = 0; i < stack.size(); i++) {
    cout << stack.front();
    tempStack.push_front(stack.front());
    stack.pop_front();
}
for (int i = 0; i < tempStack.size(); i++) {
    stack.push_front(tempStack.front());
    tempStack.pop_front();
}
```

☐
```
deque<int> tempStack;

for (int i = 0; i < stack.size(); i++) {
    cout << stack.front();
    tempStack.push_front(stack.pop_front());
}
for (int i = 0; i < tempStack.size(); i++) {
    stack.push_front(tempStack.pop_front());
}
```

☐
```
deque<int> tempStack;

while ( stack.size() > 0) {
    cout << stack.front();
    tempStack.push_front(stack.front());
    stack.pop_front();
}
while (tempStack.size() > 0) {
```

```
        stack.push_front(tempStack.front());
        tempStack.pop_front();
    }
```

☐
```
while (stack.size() > 0) {
        cout << stack.front();
        stack.push_front(stack.front());
        stack.pop_front();
    }
```

Question 19

1 pts

What is the simplest big **O** of the following time complexity?

$f(n)=5n^3+1003n^2+n.\log n+15382$

- ○ $O(n^3)$
- ○ $O(n.\log n)$
- ○ $O(5n^3)$
- ○ $O(1003n^2)$

Question 20

1 pts

What is the simplest big **O** of the following code snippet?

```
for (i = 0; i < (N / 2); ++i) {
      tempVal = userVals[i];
      userVals[i] = userVals[N - 1 - i];
      userVals[N - 1 - i] = tempVal;
}
```

- ○ O(N/2)
- ○ O(N)
- ○ O(log N)
- ○ O(3N/2)

Question 21

1 pts

What is the simplest big **O** of the following code snippet?

```
for (i = 1; i < n; i *= 2) {
    for (j = 0; j < n; j++) {
        x++;
        if (x < y) x = 4;
    }
}

for (i = 0; i < n; i++) {
    y = y * 3;
}
```

- ◯ O(n . log n)
- ◯ $O(n^2)$
- ◯ $O(n^2. \log n)$
- ◯ $O(n^3)$