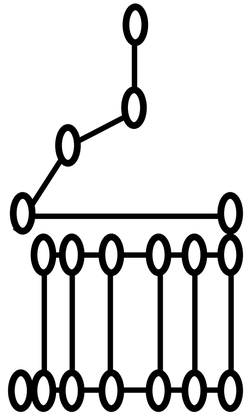
Exam 1 Review

ENGR 2730: Computers in Engineering



Exam 1 Topics

- Lecture Notes
- SVN (Reading material, Quiz)
- All Assigned Readings and Exercises
  - Chapter 6: Vectors
  - Chapter 7: Functions
  - Chapter 8: Objects and Classes
  - Chapter 9: Pointers (Basic definitions, Working knowledge)
- Questions similar to in-class clicker questions

Possible Questions for Exam 1

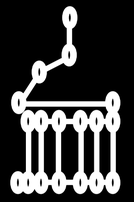
- What is a pointer?
- State the principle of least privilege.
- What is an object in the context of object-oriented programming?
- What is the purpose of the class constructor?
- What is the difference between public and private class attributes?
- Why is it better to declare class attributes (member vars) private rather than public?
- What is the difference between pass by value and pass by reference (both in theory and in practice)?
- How do data types affect results of arithmetic?

Possible Questions for Exam 1

- Define or explain data encapsulation / data hiding.
- What is a class?
- What is a member variable? What is a member function?
- When is the constructor called?
- What is a default constructor?
- Can a class have more than one constructor?
- Initializer list syntax / default values.
- How to initialize a constant member variable?
- What does it mean to instantiate an object?

Possible Questions for Exam 1

- What are the five steps of the five-step problem solving approach?
- Understand SVN commands such as: checkout, commit, ignore, add, revert, update.
- Explain what the following operators do: address, alias, de-reference.
- Explain the concept of composition.
- Know what marking a variable as `static` does.
- What are getter and setter functions and why are they used?
- Understand code of a simple constructor, setter and getter methods for a class while following the principle of least privilege.
- Understand function/constructor overloading.



CQ: What is the output of the following code segment?

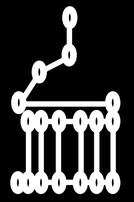
```
int i = 100;
int x = 0;

while (i >= 0)
{
    x = x + 1;
    i--;
}

cout << x << ", " << i <<endl;
```

A. 100, -1

B. 101, -1

C. 100, 0

D. 101, 0



CQ: Which of the following functions below correctly implements finding the minimum value in an array of integers?

```
int findMinArrayValueA(int arr[], int n)
{
    int minVal = -1;

    if (n > 0)
    {
        minVal = arr[0];
        for (int i = 1; i < n; i++)
        {
            if (arr[i] < minVal)
            {
                minVal = arr[i];
            }
        }
    }
    return minVal;
}
```

A

```
int findMinArrayValueB(int arr[], int n)
{
    int minVal = 1000;

    if (n > 0)
    {

        for (int i = 0; i < n; i++)
        {
            if (arr[i] < minVal)
            {
                minVal = arr[i];
            }
        }
    }
    return minVal;
}
```

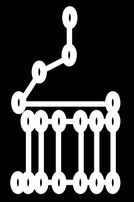
B

A. A

B. B

C. C

C: Both of the above



CQ: What does this program print out?

```
int doSomething(int a)
{
    a = a + 1;
    return 1;
}

int doSomething(float a)
{
    a = a + 1;
    return 2;
}
```

```
int doSomethingElse(int &a)
{
    a = a + 1;
    return 3;
}

int doSomethingElse(float &a)
{
    a = a + 1;
    return 4;
}
```

```
int main()
{
    float x = 2.0;
    int y = doSomething(x);
    int z = doSomethingElse(y);

    std::cout << x << " "
              << y << " "
              << z << std::endl;
}
```

A. 2 2 3

B. 2 2 4

C. 2 3 3

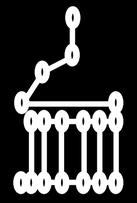
D. 2 1 3

E. 3 2 3

F. 3 2 4

G. 3 3 3

H. 3 1 3



CQ: What does this program print?

```
void addTwoA(Point2D p)
{
    p.setX (p.getX()+2) ;
    p.setY (p.getY()+2) ;
}
int main()
{
      Point2D a;
      a.setX(2);
      a.setY(2);
      addTwoA(a);
      cout<<("a.x="<<a.getX()<"a.y="<<a.getY());
}
```

```
/* class point2D is a data type to model a 2D point */
class point2D {
  public:
    void setX (float x) {m_x = x;}
    void setY (float y) {m_y = y;}
    float getX () {return m_x;}
    float getY () {return m_y;}
  private:
    float m_x; /* x coordinate of point */
    float m_y; /* y coordinate of point */
};
```

A. a.x=2.0 a.y=2.0

B. a.x=4.0 a.y=4.0

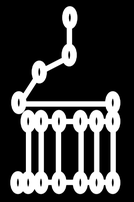
C. None of the above

CQ: Do you think having the following two function prototypes would be allowed in the same C++ program?

```
int doSomething(int a);

float doSomething(int b);
```

A. Yes

B. No



CQ: What is printed to the screen?

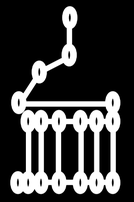
```
int mystery(int a, int &b, int c=1);

int main()
{
    int x = 2;
    int y = 2;

    int z = mystery(x,y);

    cout << x << ", " <<y << ", " << z << endl;

    return 0;
}

int mystery(int a, int &b, int c)
{
    a = a + 1;
    b = b + 1;
    c = c + 1;
    return a + b + c;
}
```

A. 2, 2, 7

B. 2, 2, 8

C. 2, 3, 7

D. 2, 3, 8



CQ: What is the output of the following program?

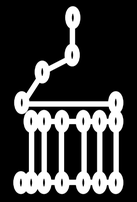
```
int main() {
    int a = 3;
    int & b = a;
    int * c = & b;

    b = 6;
    *c = 2;

    cout << "a = " << a << ", b = " << b << ", *c = " << *c << endl;

    return 0;
}
```

A. a = 3, b = 6, *c = 2

B. a = 6, b = 6, *c = 2

C. a = 3, b = 2, *c = 2

D. a = 2, b = 2, *c = 2



CQ: What is the output of the following program?

```
int main()
{
    int hist[5] = {0};
    int num_bins = 5;
    int indices[10] = {0, 0, 2, 3, 3, 3, 3, 4, 4, 4};
    int num_indices = 10;

    for (int i=0; i < num_indices; ++i)
    {
        hist[ indices[i] ]++;
    }

    for (int i=0; i < num_bins; ++i)
    {
        cout << hist[i] << " ";
    }
    cout << endl;
}
```

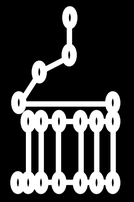
A. 0 0 0 0 0

B. 1 1 1 1 1

C. 2 0 1 4 3

D. 0 0 2 3 3

E. 2 3 4 4 4



CQ: Suppose you wish to add an additional constructor to the class defined below. Which of the following prototypes would be allowed?

```
class MyClass {
public:
    MyClass(int x=0, int y=0): m_x(x), m_y(y) {}
    void setX(int x) { m_x = x; }
    void setY(int y) { m_y = y; }
    int getX() const { return m_x; }
    int getY() const { return m_y; }
private:
    int m_x;
    int m_y;
};
```

A. MyClass(int x, int y, int z);

B. MyClass();

C. void MyClass();

D. B and C only

E. A, B and C are all valid

Problem with B : "no-parameter" constructor already defined by default parameters
Problem with C : return types (even void) not allowed for constructors

CQ: Consider the CQ class below. Which of the following member function definitions will compile?

```
class CQ {
public:
    CQ(char correctOption) : m_correctOption(correctOption)
    {
        m_other = 'A';
    }

    bool isCorrect(char option) const;

    void switchB( ) const;

    void switchC( );

private:
    const char m_correctOption;
    char m_other;
};
```

A.
```
bool CQ::isCorrect(char option) const
{
   return m_correctOption == option;
}
```

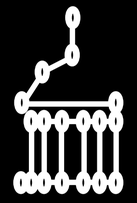
B.
```
void CQ::switchB( ) const
{
   m_other = m_correctOption;
}
```

C.
```
void CQ::switchC( )
{
    m_correctOption = m_other;
}
```

D. option A and option B.

E. option A and option C.

F. option B and option C.



CQ: What is printed as a result of calling the following function?

```
int main()
{
    float a_f = 9.0;
    float b_f = 10.0;
    int a_i = 9;
    int b_i = 10;

    cout << a_f / b_f <<", "<< a_i / b_i << ", " << b_i % a_i << endl;

    return 0;
}
```

integer division: 9/10 = 0 (truncated)

remainder after dividing 10 by 9 = 1

A. 0.900000, 0, 0

B. 0.900000, 1, 0

C. 0.900000, 0, 1

D. 0.900000, 1, 1

Five-Step Problem-Solving Methodology

1. State the problem clearly.
2. Describe the input and output.
3. Work hand examples.
4. Develop a solution/algorithm.
5. Test your solution.