Objective:

- The purpose of this quiz is to focus on the very basic fundamental concepts learned so far in previous lectures.

Question No. 1: (5.0)

What will be displayed on console when following code will be executed?

```
struct Test
{
    int a;
    Test(int i = 1)
    {
        a = i;
    }
    Test(const Test & ref)
    {
        a = ref.a;
    }
    void display()
    {
        cout << "\nNon Const Display";
    }
    void display() const
    {
        cout << "\nConst Display";
    }
    void f() const
    {
        cout << "\nin f()";
        display();
    }
    void g()
    {
        cout << "\nin g()";
        display();
    }

};
int main()
{
    Test t1;
    const Test t2(23);
    t2.display();
    t1.display();
    t1.f();
    t1.g();
    ((Test*)&t2)->a = 10;
    cout << endl << t2.a;
}
```

Run the Code

KEY

Question No. 2: (5.0)

What will be displayed on console when following code will be executed?

```
class Quiz
{
public:
    int a;
    Quiz(int i=1)
    {
        a=i;
        cout<<"\nQuiz() "<<a;
    }
    Quiz(const Quiz & ref)
    {
        a = ref.a;
        cout<<"\nQuiz (const Quiz &)
"<<a;
    }
    ~Quiz()
    {
        cout<<"\n~Quiz() "<<a;
    }
};
Quiz f(Quiz a, Quiz b, Quiz & c)
{
    static Quiz y(23);
    Quiz x(24);
    return x;
}

Quiz g1(10);
static Quiz g2(11);
Quiz g3(12);
int main()
{
    Quiz r, s(2), t(3);
    f(r,s,t);
    f(r,s,t);
}
```

Run the Code

KEY

Question No. 3: (5.0)

What will be displayed on console when following code will be executed?

```
class Time
{
    static int instanceCount;
public:
    Time()
    { instanceCount++; }
    Time(const Time&)
    { instanceCount++; }
    ~Time()
    {
      instanceCount--;
      cout<<"\n"<<Time::getInstanceCount();
    }
    static int getInstanceCount()
    { return instanceCount; }
};
int Time::instanceCount=0;
```

```
void f(Time a, Time & r)
{
   cout<<endl<<Time::getInstanceCount();
}

int main()
{
    cout<<Time::getInstanceCount();
    Time t;
    { Time t1; }
    Time t3(t);
    f(t,t3);
    cout<<Time::getInstanceCount();
    return 1;
}
```

Run the Code

KEY

Question No. 4: (4.0)

Consider below the class 'Sinlgetone'. Class 'Singleton' claims that there can be only one object of it at maximum at any given time. Is there any way to create more than one object of it in RAM? If so, then write that code in the main function provided below.

```
class Singleton
{
private:
    Singleton()
    {};
    ~Singleton()
    {}
    static Singleton * ptr;
public:
    static Singleton * CreateObject()
    {
        if (!ptr)
            ptr = new Singleton;
        return ptr;
    }
    static void freeObject()
    {
        if (ptr)
        {
            delete ptr;
            ptr=0;
        }
    }
};
Singleton * Singleton::ptr=0;

int main()
{


    Sinlgeton *firstObject = Sinlgeton::CreateObject();
    Sinlgeton secondObject(*firstObject);

    /* you can make as many objects as possible because of copy
  constructor which is implicitly provided in each class by C++. To
  eradicate this flaw, we need to explicitly define the copy constructor
  and make it private. */
```

One more challenge: I discussed that in initial weeks of semester.
I rephrased that challenge again: Is it possible at least in C++ to create/manipulate object of a class whose all contructors and destructors are private.

```
}
```