Object-oriented Programming (CS-217)
[Solution] Mid-term I Exam (Spring 2019) *(held on February 25, 2019)*
FAST-National University of Computer & Emerging Sciences (Karachi Campus)

Question 1:

a) Class & Object

A class is a data type that allows programmers to create objects. A class provides a definition for an object, describing an object's attributes (data) and methods (operations). An object is an instance of a class. With one class, you can have as many objects as required.

b) Encapsulation

Encapsulation is a process of combining data members and functions in a single unit called class. This is to prevent the access to the data directly, the access to them is provided through the functions of the class. It is one of the popular feature of Object Oriented Programming(OOPs) that helps in data hiding.

c) Access modifiers

Access modifiers set the accessibility of classes, methods, and other members. Access modifiers are a specific part of programming language syntax used to facilitate the encapsulation of components. In C++, there are only three access modifiers: Private, Public and Protected.

d)Inline function

C++ provides inline functions to help reduce function call overhead especially for small functions. The compiler places a copy of the code of that function at each point
where the function is called at compile time.

e)**this** operator

Every object in C++ has access to its own address through an important pointer called ***this*** pointer. The "***this***" pointer is an implicit parameter to all member functions.

Question 2:

a) Having multiple constructors makes a program flexible by allowing initialization of different combination of class members for different objects during their construction.

b) A setter function is still needed despite having a parameterized constructor in most situations as the constructor only "initializes" the data members during object creation. Further updates need to be performed through setter functions.

c) (1) When the function is recursive
(2) When there are loops in the function
(3) When the function contains static variables

d) **Implicit Casting:** It is performed by the compiler itself
Example:

```
int a = 10; int b = 3; float c;
c = a / b;
```

the result is saved in variable c as a float value

Explicit Casting: It is done by the programmer
Example:

```
double x = 1.2;
int y = (int) x + 1;
```

Question 3:

```
class Date
{
    int day;            // range from 1 to 31
    int month;          // range from 1 to 12
    int year;           // ranging from 2000 onwards


    // private function advance()
    void advance()
    {
        int d = day;
        int m = month;

        switch(m)
        {
            case 1:
            case 3:
            case 5:
            case 7:
            case 8:
            case 10:
            case 12:
                if(d == 31)
                {
```

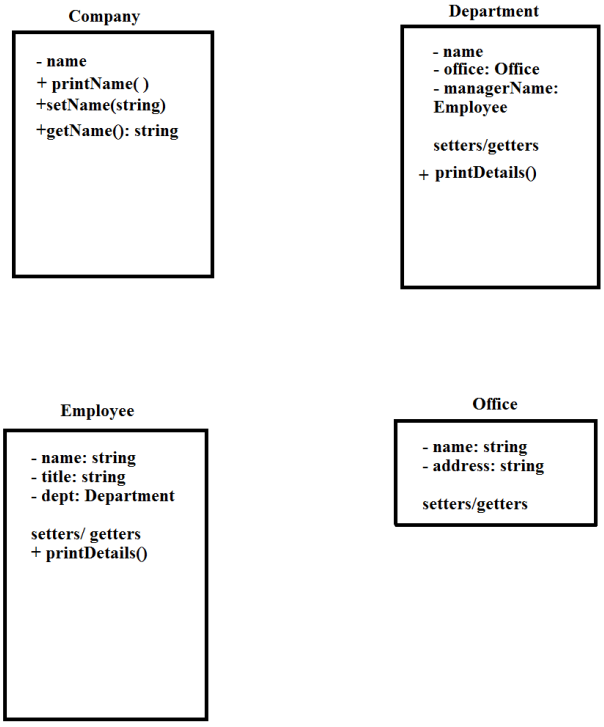
```
                        day = 1;
                        ++month;
                }
                else
                { ++day; }
                break;
        case 2:
                if(d == 29)
                {
                        day = 1;
                        ++month;
                }
                else
                { ++day; }
                break;
        case 4:
        case 6:
        case 9:
        case 11:
                if(d == 30)
                {
                        day = 1;
                        ++month;
                }
                else
                { ++day; }
                break;
    }
//year-level validation can additionally be performed
}


public:
// Date class constructor
Date(int day = 1, int month = 1, int year = 2000)
{
    if(day > 31 || month > 12 || year < 2000)
    {
        cout << "Date invalid";
        return;
    }
    this->day = day; this->month = month; this->year = year;
}

// function to set date
void setDate(int date, int month, int year)
{
    this->day = day; this->month = month; this->year = year;
}
```

```
        // function to display date
        void showDate()
        {
                cout << day << " / " << month << " / " << year << endl;
        }

        // public interface to use advance()
        void callAdvance()
        {
                advance();
        }
};

int main()
{
        Date d1(31, 1, 2000);
        d1.callAdvance();       //to call advance() for updating the date
        d1.showDate();          // display date
}
```

Question 4:

(a) Company, Department, Office & Employee

(b), (c)



d) department (const department & ob)
{

name = ob.name;
manager = ob.manager;
}

e) The departments are encapsulated within their class, while an office can exhibit composition by providing a function to create department objects. The employee class encapsulates the attributes of different employees. The access to an individual department or office's attributes can be restricted with suitable access modifiers.