1. Initialize structure

```
        struct Student{
         int num;
         char name[20];
         char gender;
         int age;
         float score;
         char addr[30];
        };
```

1st. Student stu = { 10001, “Tom",   'm',   19,   "100", “Auburn"};

2nd. struct Student{

```
         int num;
         char name[20];
         char gender;
         int age;
         float score;
         char addr[30];
        } student1={
                10001,
               “Tom",
                'm',
                19,
                "100",
                “Auburn"
           };
```

2.

```
#include<iostream>
using namespace std;
struct Date{
int month;
int day;
int year;
};
struct Student{
int num;
char name[20];
char gender;
int age;
Date birthday;
float score;
char addr[30];
};
int main(){
Student two={1,"Tom",'m',19,10,01,1993,100,"Auburn"};
Student &one=two;
one.num++;
one.birthday.day+=10;
cout<<two.num<<endl;
cout<<two.name<<endl;
cout<<two.gender<<endl;
cout<<two.age<<endl;
cout<<two.birthday.month<<endl; // output month
cout<<two.birthday.day<<endl; // output day
cout<<two.birthday.year<<endl; // output year
cout<<two.score<<endl;
cout<<two.addr<<endl;
return 0;
}
```

```
2

Tom

m

19

10/11/1993

100

Auburn
```

3. Constructor

```
class DayOfYear
        {
        public:                    □
            DayOfYear(int monthValue, int dayValue);

            void input();
            void output();
            …
        private:
            int month;
            int day;

        }
```

3.1 Based on definition of **class** DayOfYear, which declarations are legal?

DayOfYear date1(7, 4), date2(5, 5); // LEGAL!

date1.DayOfYear(7, 4);   // ILLEGAL!

date2.DayOfYear(5, 5);   // ILLEGAL!

3.2 How to initialize constructor explicitly and implicitly? (assume: variable name: date month=7 and day = 4)

Explicitly: DayOfYear date = DayOfYear (7, 4);

Implicitly: DayOfYear date(7, 4);

4 class with constructor

```
class DayOfYear
{
public:
DayOfYear(int monthValue, int dayValue);
//Initializes the month and day to arguments.


DayOfYear(int monthValue);
//Initializes the date to the first of the given month.


DayOfYear( );
//Initializes the date to January 1.


…………………………………


int getDay( );
private:
………………..
};


int main( )
{
DayOfYear date1(2, 21), date2(5), date3;
……………………
}

DayOfYear::DayOfYear(int monthValue, int dayValue)
                    : month(monthValue), day(dayValue)
{/*Body intentionally empty.*/}
```

rewrite

```
DayOfYear::DayOfYear(int monthValue, int
dayValue)
{
            month = monthValue;
            day = dayValue;
}
```

```
DayOfYear::DayOfYear(int monthValue) :
month(monthValue), day(1)
{/*Body intentionally empty.*/}
```

rewrite

```
DayOfYear::DayOfYear(int monthValue, int
day=1)
{
            month = monthValue;

}
```

```
DayOfYear::DayOfYear( ) : month(1), day(1)
{/*Body intentionally empty.*/}
```

rewrite

```
DayOfYear::DayOfYear(int month=1, int
day=1)
{                                        }
```