Day 2.

DATE

- tellg ()
  - Defined in istream class
  - Returns the posⁿ of the current character in the input stream.

- tellp ()
  - Defined in ostream class
  - Returns the position of the current character in the output stream.

- seekg ()
  To change posⁿ of get pointer in input stream.

- seekp()
  To change posⁿ of put pointer in output stream

Initializer

- Initializer list is used to initialize data member of a class.
- The list of members to be initialized is indicated with constructor as a comma separated list followed by colon

Why initializer ?

- There are situations where initialization of data members inside constructor doesn't work and initializer list must be used.
  1. non-static constant data members
  2. reference members.

DATE

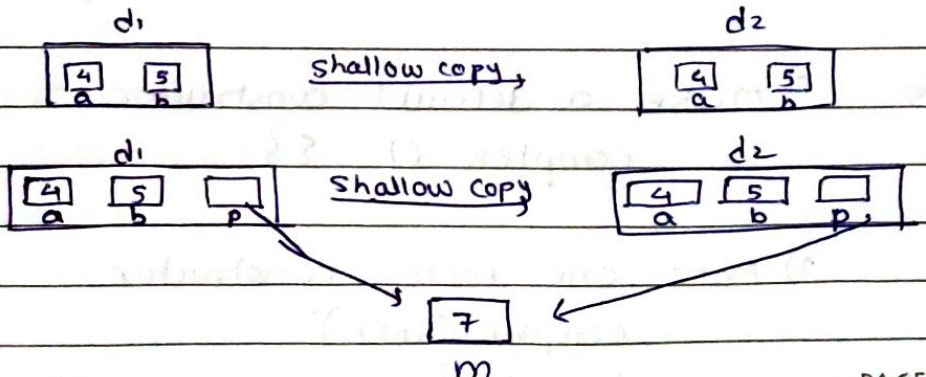
eg.

```
class dummy
{
 private:
   int a, b;
   const int x;
   int &y
 public:
   dummy (int &n) : x(5), y(n)
   { a=5, b=6; }
};
```

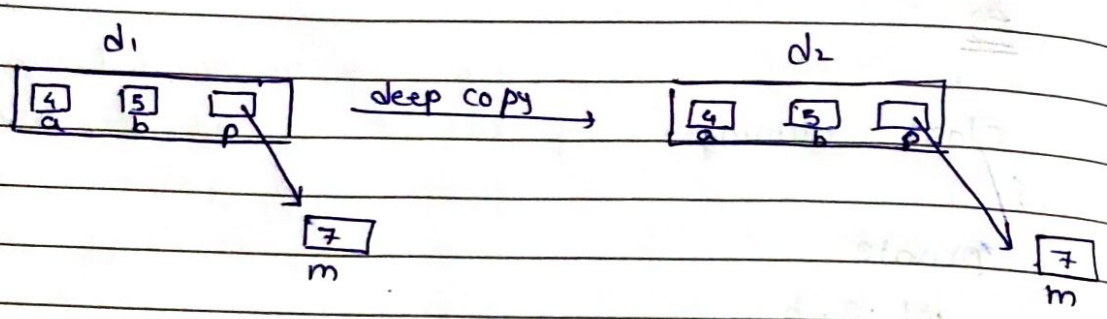
Deep copy and Shallow copy.

Shallow copy
- creating a copy of object by copying data of all member variables as it is.

Deep copy
- creating an object by copying data of another object along with the values of memory resources resides outside the object but handled by that object.



DATE



For deep copy, you need to create copy constructor as follows

```
class_name ( class_name &d )
{
    a = d.a ;   b = d.b ;
    p = new int ;
    *p = *(d.p)
}
```

Type Conversion

1) Primitive to class type.

int, char, float → any class

→ (class name)

```
Complex c1 ;
int b = 5 ;
c1 = b ;          // error
```

To resolve this error we do the type conversion.

Steps: 1) Make a default constructor.

```
complex () {}
```

2) Make one more constructor.

```
Complex (int k)
{ a = k ; }.
  b = 0 ;
```


PAGE

Day 3

DATE

2) Class type to primitive type.

```
complex c1;
c1.setdata(3,4);
int x;
x = c1          // error,
```

a
b

Class → primitive type can be implemented with casting operator.

```
operator type()
{
    return (type-data);
}
```

This will go inside class

type → int / char / float.

eg:

```
operator int()
{
    return(a);
}
```

3) Class type to class type

```
item I1;
product P1;
p1.setdata(3,4)
I1 = p1
```

item & product both are class with member variables m,n & a,b resp.

DATE

Two methods

1) Constructor - for item
2) casting operator - for product

} I1 = P1;

1) eg:

```
item () {}
item (product p)
{ m = p.a;
  n = p.b;
}
```

- Default
- ~~initi~~ converter.

If a, b are private members, use public function to access a, b.

2)

```
operator item()
{
  item temp;
  return (temp);
}
```

} inside product class