Summary

Sessions (29-12-2022)

- In Java/Python by default objects will store in heap memory. But in C++ we can store objects in the stack as well as heap memory.

```cpp
#include "iostream"
using namespace std;

class Person {
    private:
        int age;
    public:
        void setAge(int a) {
            age = a;
        }
        int getAge() {
            return age;
        }
};
main() {
    Person tom;
    tom.setAge(25);
    cout << tom.getAge() << endl;
}
```

- Above code will get space from stack memory and using "." we access the method of the object.

- We have another way to access the data or method of an object ie. using a pointer and for that we have to use the **member access operator(->)**.

.

```
main() {
    Person tom;
    Person *p;
    p = &tom;
    p -> setAge(20);
    cout << p -> getAge();
}
```

- Create a object in heap memory.

```
main() {
    Person *p = new Person;
    p -> setAge(30);
    cout << p -> getAge();
    delete p;
}
```

- In C++ we have a total 2 ways for creating an array such as static array in stack memory and dynamic array in heap memory.

- When we create a static array and if we want to extend that array then we can do but this task is very dangerous.

  For example if we create an array size of 5 and we can only store 5 elements in that array. If we try to store the 6th element in that array then it will work but as we know an array takes space from ram continuously. When they store the 6th element in that array they will overwrite the data with previous data means we lose the previous data.

- If we want to extend an array then we can extend a dynamic array only which gets space from heap memory. But if we try to extend an array like the above way then it will internally fail, it will not work as we want.

  If we want to extend then,

  1) A Memory manager allocates new continuous space in memory. For example if initially we have 20 byte of space and if we want to extend till 40 byte then memory manager will allocate new 40 byte continuous space in memory.
  2) Copying all data which is in the old memory address to the new allocated memory address means what we have in 20 byte of space that we will copy to our new 40 byte of space.
  3) Rename new address with old address.
  4) Clean old memory means 20 bytes of space.