National University of Computer & Emerging Sciences, Karachi

Fall-2019 (CS-Department)

Quiz 2

Course Code: CS-217	**Course Name:** Object-oriented Programming
Student ID:	**Section:**
Date: October 1, 2019	**Time:** 20 minutes

Question 1:

a) Briefly write the purpose of mutable variables.

b) Can we save reference to a constant variable in a constant pointer? (Yes/No)

c) Is a class with only static members really useful at all? (provide a brief but reasonable answer)

Answer:

a) A variable declared as mutable can have its value changed in a constant function.

b) Yes

c Declaring a class with only static members imply that any instance of that class won't be able to use those members, hence a class with only static members go against the principles of OOP and is least useful.

Question 2:

MajinBuu Package Delivery Service wants you to write a test version of an application for them. Whenever a package is ready for delivery, it is first sent to the Dispatch Department, which in turn forwards that package for delivery based on the receiver's Tracking ID (format: TMP-xxx), NIC# & Contact Number (format: +923xxxxxxxxx). Packages without any Tracking ID are sent back to the Vendors. Since this is a test system, your application must only allow four (4) packages in all.

Answer:

```
class Package
{
	string trackingID;
	static int pkgCount;
	public:
	Package(string trackingID = " ") {  this-> trackingID = trackingID;   pkgCount++;   }
};
int Package::pkgCount = 0;

class Dispatch_Dept
{
	public:
	void dispatch(Package pkg, Receiver r)
	{
		int count = Package::pkgCount;
		if(pkg.trackingID == " " && count < 4)
			cout << "Package returned" << endl;
		else
			send(pkg, r);
	}

	void send(Package pkg, Receiver r)
	{
		cout << "Package " << pkg.trackingID << " sent to " << r.NIC << endl;
	}
};

class Receiver
{
	string NIC;
	string contactNo;
	public:
	Receiver(string n, string c) { NIC = n; contactNo = c; }
};

int main()
{
	Package p1("TMP-111");
	Receiver r1("42301-1001010-0", "+923317777777");
	Dispatch_Dept d;

	d.dispatch(p1, r1);
}
```