“OOP কি জিনিস, কেন শিখবো, এটা দিয়ে কি করে” এই বিষয় নিয়ে লিখতে বলা হয়েছে আমাকে। প্রথমেই বলে নিই আমার এই লেখাটা একটু লম্বা হবে, তারও আগে বলে নিই, আমি Object Oriented Programming সম্পর্কে মহাজ্ঞানী কেউ না, মোটামুটি concept রাখি। আমি এই লেখাটা লিখছি নতুনদের জন্য, যারা OOP তে নতুন (ধরে নেয়া হচ্ছে যারা এটা পড়বেন তারা Structured Programming Language, যেমন C, সম্পর্কে ধারণা রাখেন)। এই লেখায়

- ভুল তথ্য থাকতে পারে

- অযৌক্তিক উদাহরণ থাকতে পারে

- ভয়াবহভাবে উদ্ভট লজিক থাকতে পারে।

তাই জ্ঞান আহরণের উদ্দেশ্যে আমার এই লেখা পড়বেন না, just বোঝার জন্য, concept clear করার জন্য এই লেখাটা। Object Oriented Programming কি, সেটা বুঝতে পারবেন, যদি মনোযোগ দিয়ে পড়েন।

============================

OOP কি? OOP আর কিছুই না, structured programming এর সাথে বাড়তি কিছু ফিচার যোগ করে একটা modified রূপ, কিন্তু অনেক strong, অনেক secured, অনেক flexible. (আমি আমার নিজের মতামত বললাম, আমার কাছে এটাই OOP, কারো ভিন্নমত থাকলে জানাতে ভুলবেন না)

আমি যতদূর জানি OOP’র আসল উদ্দেশ্য বাস্তব জীবনের সাথে সামঞ্জস্য রেখে প্রোগ্রাম করতে পারা। Structured Programming (যেমন C) –এ আমরা যা ইচ্ছা তাই করতে পারি, কোনরকম সমস্যা ছাড়াই, কিন্তু OOP তে আমরা একই কাজ আরো সহজবোধ্যভাবে করতে পারি, বাস্তব জীবনের সাথে মিলিয়ে নিয়ে। “বাস্তব জীবনের সাথে মিলিয়ে” মানে কি, সেটা একটু পরেই বুঝতে পারবেন।

দেখা যাক OOP তে আমরা কি কি feature পাচ্ছিঃ

1. All the features of Structured Programming

2. Encapsulation

3. Inheritance

4. Polymorphism etc….

আমি encapsulation, inheritance এবং polymorphism এর মানে-টা ব্যাখ্যা করছি, বাস্তব উদাহরণের সাহায্যে। আশা করি এই তিনটা সুবিধার কথা জানতে পারলেই OOP কেন শিখবো আর এটা দিয়ে কি হয়, বোঝা যাবে।

**Encapsulation** বলতে বুঝানো হয় একটা প্যাকেজিং। Structured Programming এ আমরা structure পড়েছি, এককথায় structure কে বলা যায় একাধিক ভ্যারিয়েবলের একটা প্যাকেট। ধরা যাক, যদি আমি একটা integer কাঁচাগোল্লা, float কালোজাম, আর char মন্ডা নিয়ে একটা প্যাকেট বানাই, তাহলে সেটা হবে একটা কাঁচাগোল্লা, কালোজাম আর মন্ডার একটা প্যাকেট, যেটাকে আমরা “মিষ্টান্ন” নাম দিতে পারি।

কিন্তু এই প্যাকেটে কোন security নেই, যে কেউ যখন তখন চুরি করে মিষ্টি খেয়ে ফেলতে পারে। OOP তে encapsulation আমাদেরকে ensure করে যে কেউ চুরি করে মিষ্টি খেয়ে ফেলতে পারবে না, কিংবা চট করে মিষ্টি বদলে দিতে পারবে না।

OOP তে class তৈরি করা যায়, যেভাবে Structured Programming এ আমরা structure তৈরি করতে পারতাম। Class type এর Object তৈরি করা যায়, যেভাবে আমরা struct type এর variable declare করতে পারতাম। কিন্তু class এর ভিতরে function ও রাখা যায়, যেটা আমরা structure এর ভিতরে রাখতে পারতাম না (বিকল্প পদ্ধতিতে রাখা যায়, তবে pointer ব্যবহার করে, এককথায় বলা যায় native support নেই)। এই class এর ভিতরে member variable এবং function গুলোকে private, protected কিংবা public ভাবে define করে দেয়া যায়, যেটা আমরা structure এ পারি না। class এর ভিতরে member function/variable গুলোকে নিয়ে এই নিরাপত্তা-বলয়টাকে বলা হয় encapsulation.

উদাহরণ দেই। ধরা যাক আমার বাসায় একজন লোক এসেছে। সে আমার সাথে দেখা করতে চায়। সে আমার সাথে দেখা করতে পারবে। যদি আমার বাবার সাথে দেখা করতে চায়, তাহলেও পারবে। সমস্যা হবে তখনই, যখন সে আমার বোনের সাথে দেখা করতে চাইবে। তখন আমি, আমার বড় ভাই কিংবা আমার বাবা তার কাছে কারণ জিজ্ঞাসা করবো, কারণ যদি মনমত হয়, তাহলে দেখা করতে দিবো, নয়তো দেখা করতে দিবো না। এখানে আমি, বাবা কিংবা ভাইয়া, আমরা হচ্ছি public member function, যাদেরকে সবাই access করতে পারবে। আমার বোন হচ্ছে private variable, যার সাথে দেখা করতে হলে আমাদের অর্থাৎ public member function দের কৈফিয়ত দিতে হবে। তবে public member function গুলোতে argument দিয়ে private member variable access করা যায়, যদি সেই function গুলো সেটা allow করে, এবং argument টা যদি appropriate হয়। যেমন ডাকপিয়ন এসে যদি আমার বোনের নামে একটা পার্শেল দেয়, সেটা আমার কিংবা বাবার মাধ্যমে আমার বোনের কাছে পৌঁছাবে। বুঝতেই পারছেন, এলাকার একটা বখাটে ছেলে এসে প্রেমপত্র ধরিয়ে দিলে সেটা আমি আমার বোনের কাছে দিবো না

**এবার আসি inheritance এ**। inheritance এর concept টাকে বলা যায় অনেকটা উত্তরাধিকারসূত্রে পাওয়ার মত। যেমন বাবার সম্পত্তি ছেলে উত্তরাধিকার সূত্রে পায়, তেমনি। OOP তে একটা class কে base class ধরে নিয়ে সেটা থেকে নতুন class derive করা যায়, সেক্ষেত্রে base class এর গুণাগুণ/বৈশিষ্ট্যগুলো সব derived class এর ভিতরে থাকে।

যেমন আমি যদি একটা class বানাই polygon (বহুভুজ) নামে, যেটাতে বিন্দুগুলোর স্থানাঙ্ক এবং দৈর্ঘ্য-প্রস্থ ইত্যাদি দেয়ার option আছে। তারপর যদি আমি rectangle (চতুর্ভুজ) এবং triangle (ত্রিভুজ) নামে আরো দুটো class derive করি polygon থেকে, তাহলে rectangle এবং triangle নামের class দুটোর মধ্যে বিন্দুর স্থানাঙ্ক এবং অন্যান্য property দেয়ার option চলে আসবে।আসলে কেন যেন মনের মত উদাহরণ বের করতে পারছি না inheritance এর জন্য, যদি এটুকুতে বুঝতে সমস্যা হয়ে থাকে, তাহলে এই লিংকটা একটু দেখতে পারেনঃ[http://www.cplusplus.com/doc/tutorial/inheritance/](http://l.facebook.com/l.php?u=http%3A%2F%2Fwww.cplusplus.com%2Fdoc%2Ftutorial%2Finheritance%2F&h=4AQH_qX6v&s=1)

**Polymorphism** এর জন্য হাসিন হায়দার ভাইয়ের একটা উদাহরণ থাকতে আমি নতুন উদাহরণ দিতে যাবো না, হাসিন ভাইয়ের উদাহরণটা তুলে ধরছিঃ

“অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিংয়ে পলিমর্ফিজম মানে হল একই অবজেক্টের নানা রকম রূপ পাওয়ার ক্ষমতা। সহজ বাংলায় উদাহরণ দিতে গেলে ধরেন "স্তন্যপায়ী" প্রানী একটা অবজেক্ট। এখন মানুষ স্তন্যপায়ী, কুকুর বিড়ালও স্তন্যপায়ী আবার ডলফিনও স্তন্যপায়ী। এরা প্রত্যেকে স্তন্যপায়ীর কিছু সাধারন বৈশিষ্ট্য বহন করছে যে তারা মায়ের দুধ খেয়ে বড় হয়, এরা প্রত্যেকে বাচ্চা প্রসব করে - কেউই ডিম দেয় না। কিন্তু যদি বলে ডাক দিতে, এরা একেকজন একেক আচরণ করে। আমরা আমাদের মত কথা বলে ডাকি, বিড়াল, কুকুর তাদের মত করে - এখানে কারোর সাথে কারোর মিল নেই। অথচ এরা সবাই স্তন্যপায়ী।

ঠিক একই ভাবে প্রোগ্রামিংয়ে যখন কয়েকটি অবজেক্ট একটা সাধারণ ইন্টারফেস ইমপ্লিমেন্ট করে বা একটি কমন অবজেক্ট এক্সটেন্ড করে তৈরী হয়, সেই ঘটনা কেই বলা হয় পলিমর্ফিজম।"

==================================================

যে তিনটা সুবিধার কথা বলা হলো, তার একটাও structured programming এ নেই। আশা করি বুঝতে অসুবিধা হচ্ছেনা OOP কেন শিখতে হবে।

এটা দিয়ে কি হবে? উপরে যে সুবিধাগুলোর কথা বললাম, এগুলো দিয়ে বাস্তব জীবনের সাথে মিলিয়ে program করার সুযোগ অনেক বেশি, তাই একটা ব্যাংকিং সফটওয়্যার তৈরি করতে গেলে আমি অবশ্যই OOP ব্যবহার করবো। একটা racing game তৈরি করতে গেলে আমি অবশ্যই OOP ব্যবহার করবো, কারণ অনেকগুলো গাড়ি থাকবে যাদের সবার কিছু কমন বৈশিষ্ট্য আছে, যেখানে আমি polymorphism ব্যবহার করতে পারবো।

যদি পুরো লেখাটা মনোযোগ দিয়ে পড়ে থাকেন, তাহলে “OOP কি, কেন শিখবো আর এটা দিয়ে কি হবে” এই প্রশ্নের উত্তর এতক্ষণে পেয়ে যাবার কথা। তারপরও কোন অংশে অস্পষ্টতা থাকলে এই গ্রুপ তো আছেই।