সহজ ভাষায় প্রোগ্রামিং সি/সি$^{++}$

আহমদ ওয়াহিদ মাসুদ

লুপিং নিয়ে আগের দুটি সংখ্যায় আলোচনা করা হয়েছে। শুধু কন্ডিশন এবং লুপিং পদ্ধতিতে প্রায় সব ধরনের সমস্যা সি দিয়ে সমাধান করা যায়, আর তাই সি-তে লুপিংয়ের গুরুত্ব এত বেশি।

সি-তে কয়েক ধরনের লুপিংয়ের পদ্ধতি আছে, যার কয়েকটি নিয়ে আগের সংখ্যাগুলোতে আলোচনা করা হয়েছে। এবার লুপিংয়ের বাকি পদ্ধতি নিয়ে আলোচনা করা হয়েছে।

do while সেস্টেমেন্ট : সি-তে লুপিংয়ের আরেকটি পদ্ধতি হলো do while ব্যবহার করা। এটি একটি স্টেটমেন্ট এর মাধ্যমে একটু ভিন্নভাবে লুপিং করা হয়। অনেক সময় এমন কিছু সমস্যা তৈরি হতে পারে যেগুলো প্রথমে একবার সম্পাদন করার পর পাওয়া ফলাফলের ওপর ভিত্তি করে নির্ধারণ করা হয় যে, আবার তা সম্পাদন করা হবে কি না। এ ধরনের কাজ করার জন্য do while স্টেটমেন্ট ব্যবহার করা হয়। প্রকৃতপক্ষে এটি while লুপিংয়ের আরেক রূপ। এটি নিয়ে বিস্তারিত আলোচনা করার আগে একটি ছোট উদাহরণ দেয়া হলো।

```
    printf(Òpress any key to print and ÕqÕ
to quit:\nÓ)
    char ch;
    do
    {
            ch=getch();
            printf(Ò%c\nÓ,ch);
    }while(ch!=ÕqÕ)
```

উপরের খুবই সাধারণ একটি প্রোগ্রামে ইউজারকে বলা হয়েছে q ছাড়া যেকোনো লেটার ইনপুট দিতে এবং q ইনপুট দিলে প্রোগ্রাম বন্ধ হয়ে যাবে। ওপরের প্রোগ্রামটি যদি সাধারণ while লুপ দিয়ে করা হতো, তাহলে ইউজার যেই লেটার ইনপুট দেবে মনিটরে তাই প্রিন্ট হবে। কিন্তু q ইনপুট হিসেবে দিলে প্রোগ্রাম সাথে সাথে বন্ধ হয়ে যাবে। কিন্তু এমন যদি হয় যে, ইউজার q ইনপুট দিলে তা প্রিন্ট করার পর প্রোগ্রাম বন্ধ হয়ে যাবে, তাহলে তা শুধু while লুপ দিয়ে হতো না। কারণ while লুপে প্রোগ্রাম ঢোকার সময়ই চেক করবে, ইনপুট কি q ছাড়া অন্য কোনো লেটার কি না। অন্য কোনো লেটার হলে তা লুপে ঢুকবে এবং প্রিন্ট করবে। আর ইনপুট q হলে প্রোগ্রাম লুপে ঢুকবেই না। এজন্য do while লুপ ব্যবহার করা হয়েছে। এ লুপের বৈশিষ্ট্য হলো প্রোগ্রাম প্রথমে কন্ডিশন চেক না করেই লুপে ঢুকবে। তারপর লুপের ভেতরে যা যা কমান্ড দেয়া আছে, তা এক্সিকিউট করবে। তারপর প্রোগ্রাম চেক করবে কন্ডিশন সত্য কি না, সত্য হয় তাহলে লুপ আবার এক্সিকিউট হবে আর যদি মিথ্যা হয় তাহলে প্রোগ্রাম লুপ থেকে বের হয়ে যাবে। অর্থাৎ প্রোগ্রাম শুধু প্রথমবার লুপটি চালানোর জন্য কোনো কন্ডিশন চেক করবে না, কিন্তু পরের প্রত্যেকবার চালানোর জন্য কন্ডিশন চেক করবে। যদিও এই প্রোগ্রামটি শুধু while লুপ দিয়ে চালানো সম্ভব, কারণ ওপরের সমস্যাটি সাধারণ একটি সমস্যা। কিন্তু কখনো কখনো এমন সমস্যা হতে পারে যার জন্য do while লুপ ব্যবহার করাই সব থেকে ভালো।

for loop : সি-তে for লুপকে সবচেয়ে গুরুত্বপূর্ণ লুপিং কমান্ড হিসেবে বিবেচনা করা হয়। কেননা for লুপের ব্যবহার সবচেয়ে বেশি, এ ছাড়া আর কিছুই নয়। for লুপ নিয়ে আলোচনা করার আগে আগের মতো আরও একটি প্রোগ্রাম উদাহরণ হিসেবে দেয়া হলো।

```
int i=0;
for(i=0;i<10;i++)
{
        printf(Ò%d\nÓ,i);
}
```

এটি for লুপ দিয়ে লেখা একটি প্রোগ্রাম। প্রোগ্রামটি দেখলে ধারণা করা যায়, for লুপের কাজ কী। আমরা এর আগে যত লুপের কাজ দেখেছি তাতে একটা জিনিস স্পষ্ট যে লুপ কতবার ঘুরবে তা প্রোগ্রাম আগে থেকে জানত না। প্রতিবার লুপে ঢোকার পর প্রোগ্রাম চেক করত যে কন্ডিশন সত্য না মিথ্যা, মিথ্যা হলে লুপ থেকে বের হয়ে আসত। কিন্তু for লুপ এমন একটি লুপ যাতে প্রোগ্রামকে আগে থেকে বলে দেয়া হয় লুপটি কতবার ঘুরবে। এক্ষেত্রেও প্রোগ্রামকে বারবার চেক করতে হয় প্রদত্ত শর্তটি সত্য না মিথ্যা, এক্ষেত্রে ইউজারকে অনেক কম কোড লিখতে হয় এবং এই লুপ দিয়ে অনেক সহজে কন্ডিশনিং, ইনক্রিমেন্ট/ডিক্রিমেন্টকরা যায়।

এবার for লুপের সিনটেক্স নিয়ে আলোচনা করা হয়েছে। এর সাথে ওপরে প্রদত্ত for লুপটি মিলিয়ে নিলে সহজে এর কাজ করার প্রক্রিয়া বোঝা যাবে। for লুপের কিওয়ার্ড হলো for এবং এটি লেখার নিয়ম হলো :

for (initialization; condition; increment/decrement)

for লুপের ভেতরে তিনটি অংশ। এগুলো হলো ইনিশিয়ালাইজেশন, কন্ডিশন এবং ভেরিয়েবলের মান কমানো বা বাড়ানো অর্থাৎ ইনক্রিমেন্ট বা ডিক্রিমেন্ট। লুপের কন্ডিশন চেক করার জন্য এক/একাধিক ভেরিয়েবলের প্রয়োজন। সে সব ভেরিয়েবলের লুপের প্রথম অংশে ডিক্লেয়ার করা বা ইনিশিয়ালাইজ করা হয়। প্রোগ্রামের শুরুতে ভেরিয়েবল ডিক্লেয়ার না করে for লুপের প্রথম অংশে ডিক্লেয়ার করলেও হয়। দ্বিতীয় অংশের কাজ হলো কন্ডিশন চেক করা। এখানে যেকোনো ধরনের কন্ডিশন থাকে এবং প্রোগ্রাম লুপটি প্রতিবার চালানোর সময় এই কন্ডিশন চেক করে। লুপটি একবার ঘোরার পর ওই ভেরিয়েবলের মান বাড়ানো বা কমানোর কাজ হলো তৃতীয় অংশের উদ্দেশ্য। এখানে কোড অনুসারে ভেরিয়েবলের মান পরিবর্তন করা হয়। এবার ওপরের প্রদত্ত কোডের সাথে মিলিয়ে দেখা যাক। লুপের প্রথমে একটি ভেরিয়েবল i-এর মান শূন্য করা হয়েছে। এই ভেরিয়েবলটি আগেই ডিক্লেয়ার করা হয়, চাইলে এখানেও ডিক্লেয়ার করা যেত এবং সেক্ষেত্রে i=0 না লিখে int i=0 লিখতে হতো। দ্বিতীয় অংশে কন্ডিশন হিসেবে i<10 দেয়া হয়েছে। অর্থাৎ i-এর মান যতক্ষণ না পর্যন্ত ১০ হচ্ছে ততক্ষণ লুপটি চলবে। এর পরে i-এর মান ১ করে বাড়ানো হচ্ছে অর্থাৎ i=i+1 করা হচ্ছে। আর লুপের ভেতরে প্রতিবার i-এর মান প্রিন্ট করা হচ্ছে। এবার দেখা যাক প্রোগ্রাম কিভাবে এই লুপটি নিয়ে কাজ করবে। প্রোগ্রাম প্রথমে for লুপটিতে ঢুকবে এবং i=০ করবে। তারপর প্রোগ্রাম লুপটির কন্ডিশন চেক করবে এবং দেখবে যে তা সত্য, কারণ i-এর মান এখনো ০। তাই প্রোগ্রাম লুপের ভেতরে ঢুকবে এবং ও এর মান প্রিন্ট করবে। এবার প্রোগ্রাম লুপটির তৃতীয় অংশে যাবে এবং i-এর মান বাড়িয়ে ১ করবে। তারপর প্রোগ্রাম আবার কন্ডিশন চেক করবে এবং সত্য পাবে কারণ i-এর মান এখন ১। তাই প্রোগ্রাম আবার i-এর মান প্রিন্ট করবে এবং আবার এর মান বাড়াবে। এভাবে i-এর মান ১০ না হওয়া পর্যন্ত লুপ চলতে থাকবে এবং ১-৯ প্রিন্ট হবে। তারপর i-এর মান যখনই ১০ হবে তখন প্রোগ্রাম দেখবে কন্ডিশন মিথ্যা, কারণ কন্ডিশনে বলা আছে যে i-এর মান ১০-এর থেকে ছোট হবে অর্থাৎ ১০ হওয়া যাবে না। তাই প্রোগ্রাম এখন লুপ থেকে বের হয়ে আসবে।

for লুপের ক্ষেত্রে কিছু কথা মনে রাখা ভালো। for লুপে প্রোগ্রামের ইনিশিয়ালাইজেশন অংশ শুধু একবারই এক্সিকিউট করে, কিন্তু অন্য অংশ দুটি বারবার এক্সিকিউট করে। সম্পূর্ণ লুপ একবার ঘোরাকে প্রোগ্রামিংয়ের ভাষায় iteration বলে। for লুপের তিনটি অংশ দিয়েই যে হবে এমন কোনো কথা বাধ্যবাধকতা নেই। ইউজার ইচ্ছা করলে কোনো অংশ ফাঁকাও রেখে দিতে পারেন, তবে সেক্ষেত্রে আশানুরূপ ফলাফল নাও পেতে পারেন, পরে সেদিকে খেয়াল রাখতে হবে। যেমন : ওপরের লুপকে এভাবেও লেখা যায় for (;i<10;i++)। এভাবেও কাজ করবে, কারণ ব্যবহার হওয়া ভেরিয়েবল আগেই ডিক্লেয়ার করা হয়েছে। তবে খেয়াল রাখতে হবে এভাবে ব্যবহার করলে ভেরিয়েবলকে যেনো আগে থেকে ইনিশিয়ালাইজ করা হয় অর্থাৎ ভেরিয়েবলের মান আগে থেকেই যেনো দেয়া হয়, তা না হলে ভেরিয়েবলের মান হিসেবে গারবেজ ভ্যালু থাকবে এবং লুপ ঠিকমতো কাজ করবে না। গারবেজ ভ্যালু হলো প্রোগ্রামের দেয়া ইচ্ছেমতো একটি ভ্যালু। কোডে যদি int i; লেখা হয় তাহলে প্রোগ্রাম র‍্যামের ভেতরে i নামে একটা ভেরিয়েবল ডিক্লেয়ার করে এবং এর একটা ইচ্ছেমতো ভ্যালু দিয়ে রাখে। এ কারণে for লুপে ইনিশিয়ালাইজ করা জরুরি।

ওপরের লুপকে for (i=0;;i++) এভাবেও লেখা যায়। এর মানে হলো এই for লুপে কোনো কন্ডিশন নেই। এটি একটি অসীম লুপ। আরেকটি জিনিস অবশ্যই খেয়াল রাখতে হবে for লুপের তিনটি অংশ সেমিকোলন দিয়ে আলাদা করা হয়। অংশ তিনটি লেখা হোক বা না হোক প্রতিটি অংশের জন্য আলাদা আলাদাভাবে সেমিকোলন অবশ্যই দিতে হবে। ■

ফিডব্যাক : *wahid_cseaust@yahoo.com*