সহজ ভাষায় প্রোগ্রামিং সি/সি++

আহমদ ওয়াহিদ মাসুদ

প্রোগ্রামিংয়ের জগতে সি একটি সমৃদ্ধ ল্যাঙ্গুয়েজ। এটি দিয়ে যেমন একেবারে সাধারণ প্রোগ্রাম বানানো যায়, তেমনি অনেক বড় ধরনের অ্যাপ্লিকেশন নিয়ে কাজ করার সুবিধাও এতে আছে। এ লেখায় এসব সুবিধার মধ্যে কিছু বিশেষ প্রি-প্রসেসর ডিরেক্টিভ, ম্যাক্রো ও মেমরি নিয়ে আলোচনা করা হয়েছে।

বিল্টইন ম্যাক্রো

অ্যান্সি সি-তে পাঁচটি বিল্টইন ম্যাক্রো আছে, যাদেরকে প্রয়োজনানুসারে ব্যবহার করা যায়। এগুলো হলো : __DATE__, __FILE__, __LINE__, __STDC__, __TIME__। এখানে ডেট একটি স্ট্রিং, যেখানে বর্তমান ডেট থাকে। ফাইল একটি স্ট্রিং, যেখানে যেকোনো ফাইলের নাম থাকে। লাইন একটি ইন্টিজার, যা বর্তমান লাইন নাম্বার প্রকাশ করে। এসটিডিসি চেক করে কোনো কিছু অ্যান্সি সি কি না। টাইম একটি স্ট্রিং, যেখানে বর্তমান টাইম থাকে। উল্লেখ্য, এই ম্যাক্রোগুলোকে #undef দিয়ে আনডিফাইন করা যায় না। এদেরকে ব্যবহার করাও সহজ। এর জন্য কোনো ধরনের ডিক্লারেশনের প্রয়োজন হয় না। ইউজার যেকোনো জায়গায় এদের সাধারণ ভেরিয়েবলের মতো ব্যবহার করতে পারেন। যেমন : printf(“Today is : %s”,__DATE__); । মূলত বড় কোনো অ্যাপ্লিকেশন তৈরির সময় যখন একাধিক ফাইল নিয়ে প্রোগ্রাম লিখতে হয়, তখন এই ম্যাক্রোগুলো ব্যবহার করে ডিবাগিংয়ের কাজ অনেক সহজে করা যায়। খেয়াল রাখতে হবে, এখানে ম্যাক্রোগুলোর আগে ও পরে দুটি করে আন্ডার স্কোর ব্যবহার করা হয়েছে।

কম্পাইলার কন্ট্রোল ডিরেক্টিভ

কিছু প্রি-প্রসেসর ডিরেক্টিভ আছে, যেগুলো কন্ডিশনাল কম্পাইলেশনের জন্য ব্যবহার হয়। মূলত প্রোগ্রামের পোর্টেবিলিটি বাড়ানোর জন্য এসব ডিরেক্টিভ ব্যবহার করা হয়। যেমন : #if, #ifdef, #ifndef, #elif, #else, #endif। প্রোগ্রামে ifelse যেভাবে এবং যে কারণে ব্যবহার করা হয়, এ ডিরেক্টিভগুলো ব্যবহারের ধারণাটাও অনেকটা সে ধরনের। তবে উল্লেখ্য, প্রথম পাঁচটি ডিরেক্টিভ যতক্ষণ পর্যন্ত প্রোগ্রামে #endif ডিরেক্টিভ না পাবে, ততক্ষণ পর্যন্ত কন্ডিশনাল কম্পাইল করে। একটি ছোট প্রোগ্রাম উদাহরণ দেয়া হলো :

```
int main()
{
        clrscr();
        #ifdef __STDC__
                printf(“\n Ansi C Compilance”);
        #else
                printf(“\n Not Ansi C Compilance”);
        #endif
        return 0;
}
```

ইউজারের যদি ভিন্ন ভিন্ন প্ল্যাটফমের্র জন্য কোড লিখতে হয়, তাহলেও এসব ডিরেক্টিভ ব্যবহার করে প্রোগ্রামের পোর্টেবিলিটি বাড়ানো সম্ভব। ধরা যাক, লিনআক্স ও ডসের জন্য একটি প্রোগ্রাম লিখতে হবে, যা স্যাম্পল ডিরেক্টরি থেকে টেস্ট নামের ফাইল পড়বে। তাহলে কম্পাইলার কন্ট্রোল ডিরেক্টিভ ব্যবহার করে নিচের মতো কোড লেখা যায়।

```
#ifdef __MSDOS__
        #define FILENAME “/sample/test”
#else __MSDOS__
        #define FILENAME “\\sample\\test”
#endif __MSDOS__
```

ও ## অপারেটর

এই প্রি-প্রসেসর অপারেটরগুলো অ্যান্সি সি-এর নতুন সংযোজন। অর্থাৎ ট্রাডিশনাল সি-তে এগুলো নেই। এদের মাঝে # অপারেটরকে বলে stringizing অপারেটর এবং এটি একটি ইউনারি অপারেটর, অর্থাৎ এটি একটি মেম্বারের সাথে কাজ করে। অন্যদিকে ## অপারেটরকে বলা হয় টোকেনপাস্টিং অপারেটর, যা একটি বাইনারি অপারেটর, অর্থাৎ এটি ব্যবহার করতে দুটি মেম্বার দরকার।

#-এর উদাহরণ হিসেবে একটি ছোট প্রোগ্রাম দেয়া হলো :

```
#define msg(a,b) printf(#a “and” #b “: best of luck!!”);
void main(){
        msg(romi, nipu);
}
```

stringizing অপারেটরের কাজ হলো ম্যাক্রোর আর্গুমেন্টকে স্ট্রিংয়ে রূপান্তর করা। তাই উপরের প্রোগ্রামটি যখন কম্পাইল করা হবে, তখন প্রি-প্রসেসর তা নিচের মতো অনুবাদ করে নেবে :

```
void main(){
        printf(“romi” “and” “nipu” “: best of luck!!”);
}
```

তবে এখানে স্ট্রিংগুলো যেহেতু শুধু স্পেস দিয়ে পৃথক করা আছে, তাই ওপরের প্রিন্ট ফাংশনের আর্গুমেন্টকে একটি একক স্ট্রিং হিসেবে দেখানো হবে।

এবার ##-এর উদাহরণ দেয়া হলো। এই অপারেটরের কাজ হলো দুটো টোকেনকে একত্র করা, যেমন :

```
#define N(i) n##i
N(1) = N(2) = N(3);
```

এই প্রোগ্রামকে প্রি-প্রসেসর নিচের মতো রূপান্তর করবে,

```
n1=n2=n3;
```

#প্রাগমা

এখন পর্যন্ত বিভিন্ন ধরনের প্রি-প্রসেসর ডিরেক্টিভ নিয়ে আলোচনা করা হয়েছে। তবে এই ডিরেক্টিভগুলো ছাড়া প্রতিটি কম্পাইলারের কিছু নিজস্ব ডিরেক্টিভ থাকতে পারে, যাদেরকে বলা হয় প্রাগমা। এটি ব্যবহারের নিয়ম হলো :

```
#pragma compiler_directives
```

ভিন্ন ভিন্ন কম্পাইলারের ভিন্ন ভিন্ন ডিরেক্টিভ থাকতে পারে। কোন কম্পাইলারের কী কী ডিরেক্টিভ আছে, তা সংশ্লিষ্ট কম্পাইলারের ম্যানুয়েল দেখলেই পাওয়া যাবে। যেমন : টার্বো সি-এর অনেকগুলো ডিরেক্টিভের মধ্যে দুটি হলো #pragma startup ও #pragma exit।

এখানে কোন ডিরেক্টিভ কীভাবে কাজ করে, তা জানার জন্য টার্বো সি-এর এডিটরে ডিরেক্টিভটি লিখে Ctrl+F1 প্রেস করলেই সেটি সম্পর্কে সব তথ্য পাওয়া যাবে। এই প্রাগমা ব্যবহারের নিয়ম হলো #pragma startup<function_name> ও #pragma exit<function_name>। আমরা জানি, প্রোগ্রামে লেখা সব ফাংশনই কোনো না কোনোভাবে মেইন ফাংশন থেকে কল করতে হয় এবং প্রোগ্রামের কাজও মেইন ফাংশন থেকে শুরু হয়। কিন্তু প্রাগমা ব্যবহার করে এই নিয়ম ভাঙ্গা যায়। স্টার্টআপ ডিরেক্টিভের মাধ্যমে কোনো ফাংশনকে মেইন ফাংশনের আগেই ব্যবহার করা যায়। তবে এ ক্ষেত্রে অবশ্যই ব্যবহার হওয়া ফাংশনকে প্রাগমার আগে ডিক্লেয়ার করতে হবে।

সি প্রোগ্রাম ও মেমরি

একটি সি প্রোগ্রাম রান করার সময় মেমরিকে সাধারণত চার ভাগে ভাগ করে নেয় : কোড এরিয়া, ডাটা এরিয়া, স্ট্যাক ও হিপ।

প্রোগ্রামের ইনস্ট্রাকশনগুলো কোড এরিয়াতে থাকে এবং প্রোগ্রাম যতক্ষণ চলে এই অংশ ততক্ষণ অপরিবর্তিত থাকে। উল্লেখ্য, এই অংশের সাইজ কতটুকু হবে, তা ইএক্সই ফাইল তৈরির সময়ই নির্ধারিত হয়।

ডাটা এরিয়াতে প্রোগ্রামের সব ডাটা থাকে। এই অংশকে আবার দুই ভাগে ভাগ করা যায়। যেমন : ইনিশিয়ালাইজড ও আনইনিশিয়ালাইজড। প্রোগ্রামে যেসব ভেরিয়েবলের ডাটা নির্ধারণ করা হয়, এদেরকে ইনিশিয়ালাইজড এরিয়াতে রাখা হয় এবং যাদের ডাটা নেই তাদের অপর এরিয়াতে রাখা হয়। উল্লেখ্য, প্রোগ্রাম চলার সময় এই অংশের সাইজ অপরিবর্তিত থাকে এবং এটিও ইএক্সই ফাইল তৈরি হওয়ার সময় নিজের সাইজ নির্ধারণ করে নেয়।

প্রোগ্রামের বেশিরভাগ ভেরিয়েবলের জন্য

(বাকি অংশ ৬৪ পৃষ্ঠায়)