সহজ ভাষায় প্রোগ্রামিং সি/সি++

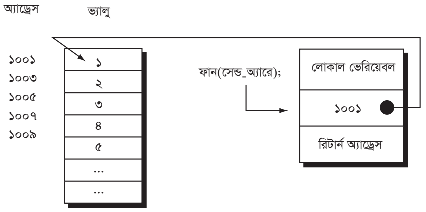
আহমদ ওয়াহিদ মাসুদ

প্রোগ্রাম ছোট হলে তাতে জটিলতা কম থাকে ঠিকই, কিন্তু তা দিয়ে জটিল সমস্যা সমাধান করা যায় না। বড় ধরনের কাজ করার জন্য দরকার বড় প্রোগ্রাম, যা কি না জটিলও হতে পারে। প্রোগ্রামে ভেরিয়েবলের গুরুত্ব কতটুকু তা আমরা জানি। প্রোগ্রামের আকার যখন বড় এবং জটিল হয়, তখন স্বভাবতই তাতে বেশি পরিমাণে ভেরিয়েবল ব্যবহার করতে হয়। ভেরিয়েবল বেশি বা কম যাই হোক না কেনো, আপনাকে খেয়াল রাখতে হবে ভেরিয়েবলগুলোর ব্যবহারের পদ্ধতি। অল্প ভেরিয়েবল যেকোনোভাবেই ব্যবহার করা যায়। কিন্তু ভেরিয়েবল সংখ্যায় বেশি হলে বিশেষ নিয়ম অনুসরণ করা উচিত।

গত সংখ্যায় অ্যারে নিয়ে আলোচনা করা হয়েছে। অ্যারে মেমরিতে কিভাবে জায়গা দখল করে এবং অ্যারের মান কিভাবে নির্ধারণ করা হয়, তা উদাহরণসহ দেখানো হয়েছে। কিন্তু অ্যারে কিভাবে ব্যবহার করা হয়, সেটি একটি গুরুত্বপূর্ণ বিষয়। ব্যবহারবিধি নিয়ে আলোচনা করার আগে আরেকবার বলে নেয়া ভালো– অ্যারে হলো অনেকগুলো ভেরিয়েবলের সমষ্টি। অর্থাৎ একই ডাটা টাইপের অনেকগুলো ভেরিয়েবলের দরকার হলে আলাদা আলাদাভাবে ডিক্লেয়ার না করে একসাথে অ্যারে ব্যবহার করা যায়। অ্যারে হাই লেভেল ল্যাঙ্গুয়েজের একটি অন্যতম বৈশিষ্ট্য।

প্যারামিটার হিসেবে অ্যারে

অ্যারে নিয়ে কাজ করলে বড় প্রোগ্রামের জটিলতা অনেক কমে যায়। কিন্তু প্রোগ্রামে ফাংশনের ব্যবহার থাকলে এমন পরিবেশের সৃষ্টি হতে পারে যে অ্যারেটি এক ফাংশন থেকে আরেক ফাংশনে পাঠানোর প্রয়োজন হলো। আমরা জানি কিভাবে ভেরিয়েবল এক ফাংশন থেকে আরেক ফাংশনে পাঠানো হয় এবং এর সুবিধা কী কী। একই সুবিধা অ্যারের ক্ষেত্রেও প্রযোজ্য।



ফাংশনের প্যারামিটার নিয়ে কাজ করতে হলে অবশ্যই ফরমাল প্যারামিটার এবং অ্যাকচুয়াল প্যারামিটার এবং ফাংশনের প্রটোটাইপ কী তা মনে রাখতে হবে। আলোচনার সুবিধার্থে প্যারামিটারগুলো সম্পর্কে সংক্ষিপ্ত ব্যাখ্যা দেয়া হলো। ফাংশন ডিক্লেয়ার করার সময় যে প্যারামিটার ব্যবহার করা হয় তাকে ফরমাল প্যারামিটার বলে। আর কোনো জায়গায় ফাংশনকে কল করার সময় যখন কোনো ভেরিয়েবল বা মান পাঠানো হয়, তখন তাকে অ্যাকচুয়াল প্যারামিটার বলে। প্যারামিটার হিসেবে অ্যারে পাঠানোর সময় মূলত খেয়াল রাখতে হয়, তা কোন প্যারামিটার হিসেবে পাঠানো হচ্ছে। আমরা জানি, অ্যারে লেখার সময় তৃতীয় বন্ধনী ব্যবহার করতে হয়। প্যারামিটারে পাঠানোর সময় এ বন্ধনী ছাড়া পাঠাতে হয়। তবে ফরমাল প্যারামিটারে আবার বন্ধনী দিতে হয়। আর প্রটোটাইপেও একইভাবে ফরমাল প্যারামিটার ব্যবহার করতে হয়। প্যারামিটারের মাধ্যমে পাঠানো একটি ছোট উদাহরণ নিচে দেয়া হলো :

```
    #include<stdio.h>
    #include<conio.h>
    void fun(int get_array[])
    {
            for(int n=0;n<5;n++)
    {
            printf("%d ",get_array[n]);
    }
    }
    int main()
    {
            int send_array[5]={1,2,3,4,5};
    fun(send_array);
    getch();
    clrscr();
    return 0;
    }
```

উপরের উদাহরণে দু'ধরনের প্যারামিটারের ব্যবহারই দেখানো হয়েছে। তবে এখানে কোনো প্রটোটাইপ দেয়া হয়নি। কারণ, ইউজার ডিফাইন্ড ফাংশনটি আগেই লেখা হয়েছে। যদি নতুন ফাংশনটি মেইন ফাংশনের পরে লেখা হতো, তাহলে মেইন ফাংশনের আগে নতুন ফাংশনটির প্রটোটাইপ দিতে হতো। লক্ষ করলে দেখা যাবে, ফরমাল ফাংশনটিতে get_array[]ভাবে অ্যারে লেখা হয়েছে। এখানে বন্ধনী দেয়া হয়েছে, যার মাধ্যমে প্রোগ্রামকে বলা হলো এই ফাংশনে যেই ভেরিয়েবল পাঠানো হবে, তা একটি অ্যারে হবে। এখানে বন্ধনীর ভেতরে কোনো মান অর্থাৎ ইনডেক্স নম্বর দেয়া হয়নি। অর্থাৎ যেকোনো সংখ্যক এলিমেন্টের অ্যারে এই ফাংশনে আসতে পারে। এখানে যদি get_array[5] লেখা হতো তাহলে প্রোগ্রামকে বলে দেয়া হতো, এই ফাংশনে যে অ্যারে আসবে তার এলিমেন্টের সংখ্যা ৫ হবে। এর কম বা বেশি হলে এরর দেখানোর সম্ভাবনা থাকবে। তাই বন্ধনীর ভেতরে ফাঁকা রাখা হয়েছে যাতে যেকোনো ইনডেক্সের অ্যারে পাঠানো সম্ভব হয়।

মেইন ফাংশনের ভেতরে প্যারামিটার হিসেবে অ্যারে পাঠানোর সময় বন্ধনী ব্যবহার করা হয়নি। কারণ, এটি অ্যাকচুয়াল প্যারামিটার। তবে ক্ষেত্রবিশেষে এখানেও বন্ধনী ব্যবহার করতে হয়। ইউজার যদি পুরো অ্যারেটিই পাঠাতে চান তাহলে অ্যাকচুয়াল প্যারামিটারে কোনো বন্ধনী দেয়া যাবে না। তবে ইউজার যদি চান পুরো অ্যারে না পাঠিয়ে অ্যারের শুধু একটি এলিমেন্ট পাঠানো হবে, তাহলে বন্ধনী ব্যবহার করতে হবে। ধরা যাক পুরো অ্যারে না পাঠিয়ে অ্যারেটির তৃতীয় এলিমেন্টটি পাঠানো হবে। তাহলে মেইন ফাংশনের ভেতরে fun (send_array[2]); এভাবে লিখতে হতো। আমরা জানি বাইনারিতে গণনা শুরু হয় শূন্য থেকে। শূন্য থেকে শুরু করলে তৃতীয় সংখ্যাটি হবে ২। এ কারণে বন্ধনীর ভেতরে ২ লিখে বোঝানো হলো তৃতীয় এলিমেন্টটি পাঠানো হলো। তবে ফরমাল এবং অ্যাকচুয়াল প্যারামিটার যদি একই না হয়, তাহলে আবার এরর দেখাবে। ইউজার যদি শুধু একক এলিমেন্ট পাঠাতে চান, তাহলে ফরমাল প্যারামিটারে অ্যারে না রেখে শুধু একটি সাধারণ ভেরিয়েবল রাখলেই চলবে। শুধু খেয়াল রাখতে হবে অ্যারের এলিমেন্ট পাঠানো হচ্ছে এবং যে ভেরিয়েবল দিয়ে তাকে গ্রহণ করা হচ্ছে, তাদের ডাটা টাইপ যেনো একই থাকে। অন্যথায় ডাটা টাইপ কনভার্সনের ক্ষেত্রে এরর বা ওয়ার্নিং দেখাতে পারে।

প্যারামিটার হিসেবে অ্যারে পাঠালে যা হয়

প্রোগ্রামে ডিক্লেয়ার করা সব অ্যারের জন্য সাইজ এবং ডাটা টাইপ অনুযায়ী প্রয়োজনীয়সংখ্যক বাইট বা জায়গা মেমরিতে দখল করা হয়। যদি কোনো অ্যারের এলিমেন্ট সংখ্যা ১০ হয় এবং প্রতিটি এলিমেন্টের জন্য ১ বাইট করে জায়গা দরকার হয় (অর্থাৎ অ্যারেটি একটি ক্যারেক্টার অ্যারে), তাহলে মেমরিতে পরপর ১০ বাইট জায়গা সংরক্ষণ করা হবে এবং সেটিই হবে ডিক্লেয়ার করা অ্যারে।

আমরা জানি, কোনো ফাংশনের আরগুমেন্ট হিসেবে ভেরিয়েবল নির্ধারণ করা হলে সি-তে সাধারণত কল বাই ভ্যালু পদ্ধতি অনুসরণ করা হয়। অর্থাৎ এক্ষেত্রে স্ট্যাকে সংশ্লিষ্ট ভেরিয়েবলের ডাটা কপি করে ফাংশনটা কল করা হয়। কিন্তু ফাংশনের প্যারামিটার হিসেবে যদি অ্যারে পাঠানো হয়, তাহলে স্ট্যাকে ওই অ্যারের সব এলিমেন্টের ডাটা কপি করা হয় না। বরং বেস অ্যাড্রেস তথা প্রথম এলিমেন্টটির অ্যাড্রেসটি রাখা হয়। পরে ওই অ্যাড্রেসের মাধ্যমেই পরের এলিমেন্টগুলোর ডাটা ব্যবহার করা হয়। উপরের উদাহরণের কথা ধরা যাক। প্রোগ্রামটি চলার সময় স্ট্যাককে চিত্র-১-এর মতো কল্পনা করা যায়। send_array[] অ্যারেটির প্রথম এলিমেন্টটির অ্যাড্রেস যদি ১০০১ হয়, তাহলে চিত্রের বাম দিকের মতো করে মেমরিতে অ্যারেটি সংরক্ষণ করা হবে। আর চিত্রের ডান দিকের মতো করে স্ট্যাকে অপারেশন হবে। স্ট্যাকে শুধু প্রথম এলিমেন্টের অ্যাড্রেস ১০০১ থাকবে। পরে অন্য কোনো এলিমেন্টের প্রয়োজন হলে এই বেস অ্যাড্রেসকে ব্যবহার করে উল্লিখিত এলিমেন্টের অ্যাড্রেস ▶

এবং এর মান বের করা হবে। তাহলে চিত্রটি দিয়ে যা বোঝানো হয়েছে তাতে এককথায় বলতে হয়, আরগুমেন্ট হিসেবে অ্যারে নির্ধারণ করা হলে স্ট্যাকে অ্যারের এলিমেন্ট কপি না হয়ে শুধু বেস অ্যাড্রেস কপি হয়।

বেস অ্যাড্রেসের মাধ্যমে অন্যান্য এলিমেন্টের অ্যাড্রেস নির্ধারণ

উপরের উদাহরণের মাধ্যমে দেখানো হলো কিভাবে স্ট্যাক অপারেশনের সময় সম্পূর্ণ অ্যারে কপি না হয়ে বেস অ্যাড্রেস কপি হয়। এখন যদি পরের এলিমেন্টগুলোর মানের দরকার হয় তাহলে প্রোগ্রাম ওই বেস অ্যাড্রেস থেকেই পরের এলিমেন্টগুলোর মান নির্ধারণ করে নেয়। বেস অ্যাড্রেস থেকে পরের এলিমেন্টের অ্যাড্রেস বের করার একটি সূত্র আছে।

Address of element [i] = base address + (I * scale factor of data type)

উপরের এই সূত্র দিয়ে শুধু বেস অ্যাড্রেসের মাধ্যমে পরের যেকোনো এলিমেন্টের অ্যাড্রেস নির্ধারণ করা হয়। এখানে স্কেল ফ্যাক্টর হলো কোনো ডাটা টাইপের জন্য প্রয়োজনীয়সংখ্যক বাইটের মান। যেমন ক্যারেক্টার টাইপ ডাটার জন্য ১ বাইট, ইন্টিজারের জন্য ২ বাইট ইত্যাদি। এই সূত্র ব্যবহার করে একটি অ্যারের এলিমেন্টের অ্যাড্রেস বের করার উদাহরণ দেয়া হলো :

ধরা যাক, int x[5] একটি অ্যারে ডিক্লেয়ার করা হলো, যার বেস অ্যাড্রেস ১০০১। তাহলে x[4] এলিমেন্টটির মেমরিতে অ্যাড্রেস হবে– Address of x[4]=base address+(4x2)=1001+8=1009। এখানে বেস অ্যাড্রেস হলো ১০০১, i-এর মান হলো ৪ এবং স্কেল ফ্যাক্টর হলো ২। কারণ এটি একটি ইন্টিজার ধরনের অ্যারে। তাই x[4] এলিমেন্টটির অ্যাড্রেস হবে ১০০৯।

ফাংশনের প্যারামিটার হিসেবে অ্যারে ব্যবহার করা হলে স্ট্যাকে ও অ্যারের বিভিন্ন এলিমেন্টের ডাটা না রেখে বেস অ্যাড্রেস রাখা হয় এবং এই অ্যাড্রেসের মাধ্যমে যেহেতু বিভিন্ন এলিমেন্ট নিয়ে কাজ করা হয়, তাই কল করা ফাংশনে কোনো এলিমেন্টের ডাটা পরিবর্তন করা হলে কলার ফাংশনেও পরিবর্তিত মানটি পাওয়া যাবে। স্ট্যাকে যদি অ্যাড্রেস না রেখে মান রাখা হতো তাহলে কল করা ফাংশনে ডাটার পরিবর্তন করা হলেও কলার ফাংশনে ডাটার কোনো পরিবর্তন হতো না। যেহেতু এখানে সরাসরি অ্যাড্রেস নিয়ে কাজ করা হচ্ছে, তাই একটির পরিবর্তনে আরেকটিও পরিবর্তন হয়ে যাচ্ছে। কারণ সবার বেস অ্যাড্রেস একই। এ পদ্ধতিকে বলা হয় কল বাই রেফারেন্স, অর্থাৎ অ্যাড্রেসের মাধ্যমে কল করা।

প্যারামিটারে মান না পাঠিয়ে ইউজার সরাসরি অ্যাড্রেসও পাঠাতে পারেন। সেক্ষেত্রে অ্যাড্রেস অপারেটর ব্যবহার করতে হবে। যেমন a দিয়ে যদি কোনো ভেরিয়েবল বোঝায় তাহলে &a দিয়ে ওই ভেরিয়েবলের অ্যাড্রেসকে সরাসরি বোঝায়। সাধারণত কোনো ডাটা ইনপুট নেয়ার সময় scanf ফাংশনের ভেতরে এ ধরনের অপারেটর ব্যবহার করা হয়। কারণ ইউজার যখন ইনপুট দেয় তখন সেই ইনপুটটি কোথায় যাবে অর্থাৎ কোন অ্যাড্রেসে রাখা হবে তা সরাসরি বলার দরকার হয়। প্যারামিটারে ইউজার যদি সরাসরি অ্যাড্রেস পাঠাতে চান, তাহলে অ্যারের এলিমেন্ট এবং এলিমেন্টের আগে এই অ্যাড্রেস অপারেটর ব্যবহার করতে হবে। উপরে দেখানো হয়েছে স্ট্যাক অপারেশনের সময় অ্যাড্রেস ব্যবহার হয়। কিন্তু সরাসরি অ্যাড্রেস ব্যবহারের কিছু সমস্যা আছে। যেমন সরাসরি অ্যাড্রেস যদি প্যারামিটার হিসেবে পাঠানো হয়, তাহলে যেকোনো এক জায়গায় মান পরিবর্তন করা হলে মূল এলিমেন্টের মান পরিবর্তন হয়ে যাবে। তাই কোনো মান যদি পরিবর্তন করা হয়, তার মানে হলো ওই অ্যাড্রেসে যেই মান অবস্থিত, তার মান সরাসরি পরিবর্তন করে দেয়া। আর মূল এলিমেন্ট এবং ইউজার ডিফাইন্ড ফাংশনের ভেতরের পাঠানো অ্যারের এলিমেন্টের অ্যাড্রেস সমান। সেটা স্ট্যাকের অপারেশনের সময় চিত্রে দেখানো হয়েছে। তাই যেহেতু অ্যাড্রেস একই, তাই যেকোনো একটির মান পরিবর্তন করলেই মূল মানটি পরিবর্তন হয়ে যাবে।

অ্যারে ব্যবহারের সময় কিছু বিষয় গুরুত্ব সহকারে খেয়াল করতে হবে। আমরা জানি অ্যারে মানে হলো অনেকগুলো ভেরিয়েবলের সমষ্টি। সেই ভেরিয়েবলগুলো আবার পরপর থাকবে, অর্থাৎ এদের মাঝে আর অন্য কোনো ভেরিয়েবল থাকতে পারবে না। এটি একদিকে যেমন ভালো, অপরদিকে তেমনি খারাপ। ভালো বলা হলো এ কারণে, পরপর ভেরিয়েবলগুলো থাকার জন্যই শুধু বেস অ্যাড্রেস দিয়ে হিসাব করে বাকি যেকোনো এলিমেন্টের অ্যাড্রেস এবং তার মান বের করা সম্ভব। তাই অ্যারেটিকে কোথাও পাঠাতে হলে শুধু বেস অ্যাড্রেস পাঠানো হয় এবং সেই অ্যাড্রেসের মাধ্যমেই প্রোগ্রাম নিজে থেকে সবকিছু নির্ধারণ করে নেয়। কিন্তু ভেরিয়েবলগুলোর প্রত্যেকটির একটি নির্দিষ্ট সাইজ আছে। ডাটা টাইপের ওপর নির্ভর করে এই সাইজ কতটুকু হবে। অ্যারে যদি ইন্টিজার হয়, তাহলে প্রতিটি এলিমেন্টের সাইজ ২ বাইট করে হবে। যদি ফ্লোটের অ্যারে নেয়া হয় তাহলে ৪ বাইট করে জায়গা নির্ধারণ করা হবে। এখন মেমরিতে সব ডাটা একসাথে থাকে না। একেক জায়গায় একেক ধরনের ডাটা থাকে। তাই মেমরিতে যে পরপর অনেকগুলো জায়গা ফাঁকা থাকবেই তার কোনো নিশ্চয়তা নেই। ধরা যাক, কোনো প্রোগ্রামে ১০০০০ এলিমেন্টের একটি ডাবল টাইপের অ্যারে ডিক্লেয়ার করার প্রয়োজন হলো। আমরা জানি ডাবল টাইপের ভেরিয়েবল সাধারণত ৮ বাইট করে নেয়। ৬৪ বিটের অপারেটিং সিস্টেম হলে ১৬ বাইট করে নেবে। এখন একটি এলিমেন্টই যদি ১৬ বাইট জায়গা দখল করে তাহলে ১০০০০ এলিমেন্টের জন্য ১৬×১০০০০ বাইটের প্রয়োজন হবে। এতে খুব একটা সমস্যাই হতো না যদি এ জায়গায়টি একসাথে না লাগত। যেহেতু এটি একটি অ্যারে, তাই এই বিশাল জায়গা একসাথে অ্যালোকেট করতে হবে। কমপিউটারের মেমরি যদি খুব বেশি না হয়, তাহলে এরকম জায়গায় এসে প্রোগ্রাম হ্যাং করবে।

অ্যারে প্রোগ্রামিং ল্যাঙ্গুয়েজের একটি গুরুত্বপূর্ণ অধ্যায়। অ্যারের জন্য অসংখ্য ভেরিয়েবল একসাথে নিয়ন্ত্রণ করা সম্ভব। অ্যারের ব্যবহার একটু খেয়াল করে না করলে প্রোগ্রাম বন্ধ হয়ে যাওয়ার সম্ভাবনা থাকে। তাই অ্যারে সাবধানের সাথে ব্যবহার করা উচিত কজ

ফিডব্যাক : wahid_cseaust@yahoo.com