সহজ ভাষায় প্রোগ্রামিং সি/সি++

আহমদ ওয়াহিদ মাসুদ

সি-তে ফাইল অপারেশনের জন্য অনেক ধরনের ফাংশন আছে। গত পর্বে ফাইল ওপেন করা নিয়ে আলোচনা করা হয়েছিল। এ লেখায় ফাইল সংক্রান্ত অন্যান্য ফাংশন নিয়ে আলোচনা করা হয়েছে।

ফাইল বন্ধ করা

সি-তে ফাইল নিয়ে কাজ করতে হলে প্রথমে সেই ফাইল ওপেন করতে হয়। তবে কাজ শেষে সেই ফাইল বন্ধ করা উচিত। ফাইল বন্ধ করার জন্য সি-তে বিল্টইন ফাংশন দেয়া আছে। নিচে একটি উদাহরণে তা দেখানো হলো :

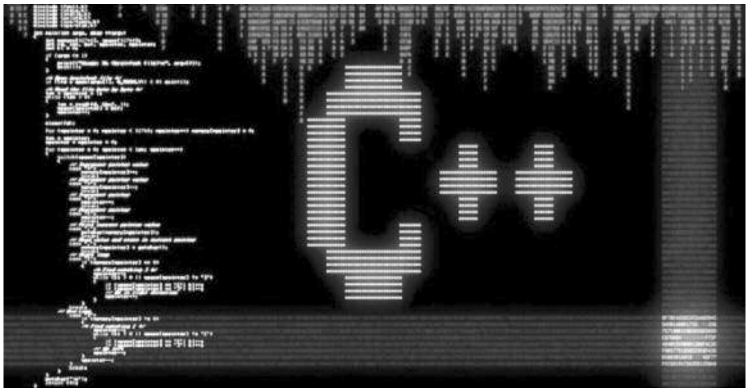
```
prototype: fclose(file_pointer)
header:         stdio.h
FILE *x;
x=fopen("temp.txt","w");
.... ....
.... ....
fclose(x);
```

এখানে প্রথমে x নামে একটি ফাইল পয়েন্টার ডিক্লেয়ার করা হয়েছে এবং পরে তা দিয়ে টেম্প নামে একটি টেক্সট ফাইল রাইট মোডে ওপেন করা হয়েছে। পরে কিছু কাজ করার পর তা fclose() ফাংশনের মাধ্যমে বন্ধ করা হয়েছে। উল্লেখ্য, ফাইল বন্ধ না করলেও প্রোগ্রাম কাজ করবে। তবে এটি খুবই ঝুঁকিপূর্ণ, কারণ প্রোগ্রাম শেষ হওয়ার পরও পয়েন্টার ওই ফাইলকে ঠিকই পয়েন্ট করে থাকে। সুতরাং পরবর্তী সময়ে ওই ফাইলের ডাটা নষ্ট হওয়ার সম্ভাবনা থাকে। যদিও এ ধরনের সমস্যা যাতে না হয়, তাই আধুনিক উইন্ডোজে কোনো প্রোগ্রাম বন্ধ হওয়ার সাথে সাথে ফাইলগুলোও অপারেটিং সিস্টেম নিজে থেকেই বন্ধ করে দেয়। কিন্তু এরপরও ঝুঁকি না নিয়ে প্রোগ্রামারের উচিত ফাইল বন্ধ করার কোড লেখা। কেননা, কোনো কারণে যদি অপারেটিং সিস্টেম ফাইল বন্ধ করতে না পারে, তাহলে ফাইলের ডাটা নষ্ট হওয়ার একটি বড় সম্ভাবনা থাকে।

ফাইলের ইনপুট/আউটপুট অপারেশন

এতক্ষণ কীভাবে ফাইল খোলা ও বন্ধ করা যায়, তা দেখানো হলো। নিচে ফাইলে ডাটা লেখা ও ফাইল থেকে ডাটা রিড করার পদ্ধতি দেখানো হয়েছে। ফাইলের ডাটা নিয়ে কাজ করার জন্যও সি-তে লাইব্রেরি ফাংশন আছে। যেমন : putc(), getc(), putw(), getw(), fputs(), fgets(), fprintf(), fscanf(), fwrite(), fread()।

কোনো ফাইলে একটি ক্যারেক্টার লিখতে putc() ও একটি করে ক্যারেক্টার পড়তে getc() ফাংশন ব্যবহার করা হয়। ক্যারেক্টার লেখার প্রটোটাইপ হলো putc(c, fp)। এখানে c হচ্ছে যে ক্যারেক্টারটি ফাইলে লিখতে হবে। আর fp হলো যে ফাইলে লিখতে হবে তার পয়েন্টার। একইভাবে getc()-এর প্যারামিটারে শুধু পয়েন্টার দিলেই হবে। উল্লেখ্য, getc() ফাংশনটি ফাইলের EOF না পাওয়া পর্যন্ত ক্যারেক্টার পড়তে থাকবে। EOF অর্থ হলো এন্ড অফ ফাইল। এটি প্রতিটি ফাইলের শেষে থাকে। এটি একটি সিম্বল, যা দিয়ে কোনো ফাইলের সমাপ্তি বোঝানো হয়। এছাড়া putc() ফাংশনও BIGd না পাওয়া পর্যন্ত কিবোর্ড থেকে ক্যারেক্টার রিড করতে থাকে। আর এওএফ বোঝানোর জন্য ইউজারকে Ctrl+D বা Ctrl+Z চাপলেই হবে।



ফাইলে কোনো স্ট্রিং লেখা বা পড়তে সাধারণত fputs() ও fgets() ফাংশন ব্যবহার করা হয়। fputs() ফাংশনের ব্যবহার একদমই সহজ। ফাংশনের প্যারামিটার হিসেবে ইনপুট স্ট্রিং দিয়ে তারপর কমা দিয়ে ফাইলের নাম দিলেই হবে। আর ইনপুট স্ট্রিং হিসেবে ইউজার কোন স্ট্রিং ভেরিয়েবল ব্যবহার করতে পারেন অথবা চাইলে ডাবল কোটের মাঝে সরাসরি একটি স্ট্রিংও লিখতে পারেন। আর প্রোগ্রাম নাল ক্যারেক্টার না পাওয়া পর্যন্ত স্ট্রিং নিতে থাকে। তাই সরাসরি স্ট্রিং ইনপুট দিতে হলে নাল ক্যারেক্টারও শেষে দিয়ে দিতে হয়। প্রোগ্রাম যদি সফলভাবে স্ট্রিংটি নিতে পারে, তাহলে তা ফাইলে লেখা হবে। আর কোনো এরর ঘটলে ফাংশনটি BIGd রিটার্ন করবে। অন্যদিকে fgets() ফাংশনের জন্য তিনটি প্যারামিটার ব্যবহার করতে হয়। প্রথম প্যারামিটারটি হলো একটি স্ট্রিং ভেরিয়েবল। কোনো ফাইল থেকে যে স্ট্রিং পড়া হবে, তা এই স্ট্রিং ভেরিয়েবলে সেভ হবে। দ্বিতীয় প্যারামিটার হলো একটি ধনাত্মক সংখ্যা n। এটি দিয়ে বোঝায় ফাংশনটি ফাইলের শুরু থেকে n-1তম ক্যারেক্টার পড়ে তা স্ট্রিং ভেরিয়েবলের মাঝে সেভ করবে। তবে n-1তম ক্যারেক্টার পাওয়ার আগেই যদি ফাইলে নিউ লাইন (\n) পাওয়া যায়, তাহলে শুধু ওই নিউ লাইন পর্যন্তই স্ট্রিং হিসেবে সেভ করা হবে। আর শেষ প্যারামিটার হলো যে ফাইল থেকে স্ট্রিং পড়তে হবে, সেই ফাইলের নাম। যদি এই ফাংশনটি ঠিকমতো কাজ করে, তাহলে তা স্ট্রিং ভেরিয়েবলের অ্যাড্রেস রিটার্ন করবে। অন্যথায় তা BIGd পেলেও নাল রিটার্ন করবে। নিচে উদাহরণ হিসেবে একটি প্রোগ্রাম দেয়া হলো :

```
FILE fp;
string getline="";
fp=fopen("test.txt","w");
fputs("This is a string",fp);
fclose();
fp=fopen("test.txt"."r");
fgets(getline,21,fp);
printf("%s",getline);
fclose();
```

এখানে প্রথমে একটি ফাইল পয়েন্টার এফপি ও একটি স্ট্রিং ভেরিয়েবল গেটলাইন ডিক্লেয়ার করা হয়েছে। তারপর এফপি দিয়ে ফাইলটিকে ওপেন করে তাতে একটি স্ট্রিং লেখা হয়েছে। এরপর ফাইলটিকে বন্ধ করে আবার ওপেন করে তা থেকে ওই স্ট্রিংটি গেটলাইন নামের একটি ভেরিয়েবলে নিয়ে তা প্রিন্ট করা হয়েছে। এখানে লক্ষ করলে দেখা যাবে, সবশেষে ফাইলটি বন্ধ করা হলেও ফাইলটি রাইট করার পরও একবার তা বন্ধ করা হয়েছে। এর কারণ সবার প্রথমে যখন এফপি পয়েন্টার দিয়ে ফাইলটি ওপেন করা হলো, তখন তা ফাইলের প্রথমে পয়েন্ট করছিল। এরপর স্ট্রিং রাইট করার পর কিন্তু পয়েন্টারটি ফাইলের শেষে পয়েন্ট করেছে। সুতরাং তখনই যদি আবার ওই একই পয়েন্টার দিয়ে ফাইল রিড করা হয়, তাহলে তা নাল রিটার্ন করবে, কারণ পয়েন্টারটি ফাইলের শেষে পয়েন্ট করে আছে। তাই এখানে ফাইলটিকে বন্ধ করে আবার ওপেন করা হয়েছে, যাতে রিড করার ▸

আগে পয়েন্টারটি ফাইলের প্রথমে পয়েন্ট করে থাকে।

fprintf() ও fscanf() ফাংশনগুলোও যথাক্রমে ফাইলে ডাটা পড়া কিংবা লেখার জন্য ব্যবহার করা হয়। এদের কাজ অনেকটা printf() ও scanf() ফাংশনের মতো। তবে পার্থক্য হলো উভয় ফাংশনের প্রথমেই আর্গুমেন্ট হিসেবে একটি ফাইল পয়েন্টার পাঠাতে হয়। এ ক্ষেত্রে যে ফাইল পয়েন্টার পাঠানো হবে fprintf(), তার আউটপুট সেই পয়েন্টেড ফাইলে লিখবে। আর fscanf() তার ইনপুট সেই পয়েন্টেড ফাইল থেকে নেবে। যেমন : fscanf(fp,"%s",name); এখানে নেম হচ্ছে ফাইলের নাম। উল্লেখ্য, এ দুইটি ফাংশনের কাজ আগের বর্ণিত ফাংশন দুইটি দিয়েও করা যায়। তবে এদের কাজ করার ধরন ভিন্ন। ইউজার আগের ফাংশন দিয়ে ফাইলে কিছু লেখার পর ওই ফাইলটি আলাদাভাবে ওপেন করলে দেখা যাবে ফাইলের মাঝে যা ইনপুট দেয়া হয়েছে, তা-ই লেখা আছে। কিন্তু পরের ফাংশন দিয়ে লেখার পর যদি ওই ফাইলটিকে আলাদাভাবে ওপেন করা হয়, তাহলে দেখা যাবে গারবেজ ডাটার মতো দুর্বোধ্য কিছু লেখা আছে। আসলে এটি গারবেজ ডাটা নয়। বরং এটি প্রোগ্রামের নিজস্ব এনক্রিপ্ট করা কোড। সুতরাং আগের পদ্ধতিতে ফাইলে কিছু লেখার পর অন্য কেউ সহজেই তা পরিবর্তন করে দিয়ে পারে। কিন্তু পরের ফাংশন দিয়ে যেহেতু এনক্রিপ্ট করা কোড ফাইলে রাইট করা হয়, তাই সেটি কেউই আলাদাভাবে বুঝতে পারবে না। সেটি দেখার একমাত্র উপায় হচ্ছে যে প্রোগ্রাম দিয়ে ফাইলটি রাইট করা হয়েছে, সেই প্রোগ্রাম দিয়েই ফাইলটি রিড করা। অর্থাৎ এই ফাংশনের সিকিউরিটি অনেক বেশি।

র‍্যান্ডম অ্যাক্সেস

আমরা জানি কোনো ফাইলে ডাটা লেখা বা ফাইল থেকে ডাটা পড়াকে ফাইল অ্যাক্সেস বলে। ফাইল অ্যাক্সেস দুই ধরনের। একটি সিকোয়েন্সিয়াল অ্যাক্সেস, আরেকটি র‍্যান্ডম অ্যাক্সেস। উপরে যেসব ফাংশন নিয়ে আলোচনা করা হলো তার সবই সিকোয়েন্সিয়াল অ্যাক্সেস। এর অর্থ হলো, এ ক্ষেত্রে ফাইলে ডাটা লেখার সময় একটা ক্যারেক্টারের পর অন্য ক্যারেক্টার লেখা হয় এবং ডাটা পড়ার সময়ও একই নিয়মে পড়া হয়। তা এই পদ্ধতিতে ফাইলের শেষে কোনো কিছু পড়তে হলে শুরু থেকে আগে সব ক্যারেক্টার পড়তে হবে। কিন্তু কোনো প্রোগ্রামে এমনও হতে পারে, ফাইলের শুরু থেকে নয় বরং মাঝখান থেকে কিছু পড়ার দরকার। এ ক্ষেত্রে ফাইলকে র‍্যান্ডম অ্যাক্সেস করতে হবে। এই কাজের জন্য সি-তে যেসব লাইব্রেরি ফাংশন আছে, তা হলো fseek(), ftell(), rewind() ইত্যাদি।

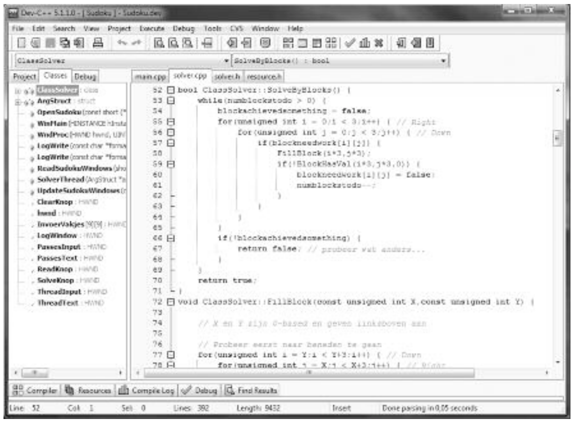
prototype: fseek(filepointer, offset, position);

ftell(filepointer);

rewind(filepointer);

এই fseek() ফাংশনের মাধ্যমে ফাইল পয়েন্টারকে ফাইলের যেকোনো স্থানে নিয়ে যাওয়া যায়। এই কাজ করার জন্য ইউজারকে ডাটা ফাইল সম্পর্কে জানতে হবে।

ডাটা ফাইলে লেখা প্রতিটি ক্যারেক্টারের অবস্থান একটি সংখ্যা দিয়ে উপস্থাপন করা হয়। প্রথম ক্যারেক্টারের জন্য ০, দ্বিতীয় ক্যারেক্টারের জন্য ১ এভাবে অবস্থান নির্ধারণ করা হয়। এই নাম্বারিং ফাইলের শেষ পর্যন্ত চলতে থাকে। আবার কোনো ক্যারেক্টারের অবস্থান ফাইল পয়েন্টারের সাপেক্ষে বের করার জন্য তার অফসেট ব্যবহার করা হয়। ফাইল পয়েন্টার যে ক্যারেক্টারে থাকে, সেই ক্যারেক্টারের অফসেট হলো ০ এবং এভাবে পরবর্তী ক্যারেক্টারগুলোর অফসেট বের করা হয়। ডাটা ফাইলের শেষ ক্যারেক্টারটি হলো ইওএফ, যা অপারেটিং সিস্টেম নিজেই দিয়ে দেয়।



```
bool ClassSolver::SolveByBlocks() {
    while(numblockstodo > 0) {
        blockachievedsomething = false;
        for(unsigned int i = 0;i < 3;i++) { // Right
            for(unsigned int j = 0;j < 3;j++) { // Down
                if(blockneedwork[i][j]) {
                    FillBlock(i*3,j*3);
                    if(!BlockHasVal(i*3,j*3,0)) {
                        blockneedwork[i][j] = false;
                        numblockstodo--;
                    }
                }
            }
        }
        if(!blockachievedsomething) {
            return false; // probeer wat anders...
        }
    }
    return true;
}
void ClassSolver::FillBlock(const unsigned int X,const unsigned int Y) {

    // X en Y zijn 0-based en geven linksboven aan

    // Probeer eerst naar beneden te gaan
    for(unsigned int i = Y;i < Y+3;i++) { // Down
        for(unsigned int j = X;j < X+3;j++) { // Right
```

fseek() ফাংশনের মাধ্যমে ফাইল পয়েন্টারকে ফাইলের কোনো নির্দিষ্ট বাইটে পাঠাতে হলে অফসেটের জন্য সেই বাইটের অফসেট মান নির্ধারণ করতে হয়। পজিশনের মাধ্যমে অফসেটের গণনার কাজ কোন জায়গা থেকে আরম্ভ হবে তা নির্ধারণ করা হয়। এখানে পজিশনের জন্য যেকোনো একটি কনস্ট্যান্ট নির্ধারণ করা যায়, SEEK_SET 0 (ফাইলের প্রথম থেকে), SEEK_CUR 1 (ফাইল পয়েন্টারের বর্তমান অবস্থান থেকে), SEEK_END 2 (ফাইলের শেষ থেকে)। যেমন কোনো ফাইলের প্রথম থেকে ১০১ নম্বর ক্যারেক্টারটি পড়ার জন্য fseek() ফাংশনকে নিচের মতো ব্যবহার করতে হবে :

fseek(fp, 101, 0); অথবা fseek(fp, 101, SEES_SET);

এ ক্ষেত্রে ফাইল পয়েন্টার প্রথম থেকে ১০১ নম্বর পজিশনে অবস্থান করবে। ফাংশনটি ঠিকমতো কাজ করলে 0 রিটার্ন করবে, অন্যথায় অন্য যেকোনো মান রিটার্ন করতে পারে। র‍্যান্ডম অ্যাক্সেস সংক্রান্ত অন্য দুইটি লাইব্রেরি ফাংশনের মাঝে ftell() ব্যবহার করা হয় ফাইল পয়েন্টারের বর্তমান অবস্থান জানার জন্য। আর rewind() ব্যবহার করা হয় ফাইল পয়েন্টারকে ফাইলের একদম শুরুতে নিয়ে যাওয়ার জন্য।

র‍্যান্ডম অ্যাক্সেসের একটি বাস্তব উদাহরণ হিসেবে mp3 ফাইলের কথা বলা যায়। এর সাথে আমরা সবাই পরিচিত। একটি mp3 ফাইলের মূল অডিও ডাটার সাথে গানের টাইটেল, অ্যালবামের নাম, আর্টিস্টের নাম ইত্যাদি কিছু তথ্যও রাখা যায়। এই তথ্যগুলোকে বলা হয় ফাইলের ট্যাগ। তবে এই ট্যাগের কয়েক ধরনের সংস্করণ আছে। কোনো সংস্করণে ট্যাগ ডাটা অডিও ডাটার আগে থাকে আবার কোনোটিতে থাকে অডিও ডাটার পরে। প্রচলিত সংস্করণ হলো ID3v1, যা এখনকার বিভিন্ন mp3 প্রোগ্রামে ব্যবহার করা হয়।

এই ভার্সনে ট্যাগের সাইজ হলো ১২৮ বাইট, যা ফাইলের শেষে অডিও ডাটার পরে থাকে। এই ১২৮ বাইটে ট্যাগ, টাইটেল, আর্টিস্ট, অ্যালবাম, সাল, কমেন্ট এবং জনরা সম্পর্কিত তথ্য থাকে। এই তথ্যগুলোর জন্য যথাক্রমে ৩, ৩০, ৩০, ৩০, ৪, ৩০ এবং ১ বাইট ব্যবহার হয়। সুতরাং ইউজার যদি এসব ডাটা অ্যাক্সেস করতে চায়, তাহলে সহজ উপায় হলো র‍্যান্ডম অ্যাক্সেস। প্রথমে পয়েন্টারকে ফাইলের একদম শেষে নিয়ে যেতে হবে, তারপর উপরে বর্ণিত বাইটের জায়গা ও সিকোয়েন্স অনুযায়ী পয়েন্টারকে পেছনে নিয়ে যেতে হবে।

সি-তে ফাইল নিয়ে এ ধরনের কাজ করার জন্য প্রচুর লাইব্রেরি ফাংশন রয়েছে, যেগুলোর বেশিরভাগই stdio.h ও io.h-এ বর্ণিত। কোনো হেডার ফাইলে কী কী ফাংশন আছে আর তাদের প্রত্যেকের কাজ কী, তা আসলে বলে শেষ করা যাবে না। এসব ফাংশন সম্পর্কে জানতে চাইলে টার্বো সি-এর হেল্প ফাইলের সাহায্য নেয়া যেতে পারে কিংবা কোনো রেফারেন্স বই ব্যবহার করা যেতে পারে। সি ল্যাঙ্গুয়েজের জন্য রেফারেন্স হিসেবে সবচেয়ে বেশি প্রচলিত বই হলো হার্বার্ট শিল্ডের টার্বো সি : দ্য কমপিউটার রেফারেন্স। আর টার্বো সি-এর হেল্প ব্যবহার করতে হলে এডিটরে ফাংশনটি লিখে ফাংশনের প্রথমে কার্সর বা মাউস পয়েন্টার রেখে Ctrl+F1 চাপলে অথবা ফাংশনের ওপর ডাবল ক্লিক করলে সংশ্লিষ্ট ফাংশন কীভাবে কাজ করে সে সংক্রান্ত ডাটা স্ক্রিনে দেখাবে। টার্বো সি-এর হেল্প ফাইলে বেশিরভাগ ফাংশনেরই বর্ণনা ও উদাহরণ দেয়া আছে। তাই কোন ফাংশন কীভাবে কাজ করে, তা বোঝার জন্য উদাহরণটুকু কপি করে এনে নিউ ফাইলে লিখে রান করালেই হবে।

সি ল্যাঙ্গুয়েজ দিয়ে করা যায় না এমন ফাইল সংক্রান্ত কাজ কমই আছে। যদিও এখনকার আধুনিক ল্যাঙ্গুয়েজগুলোতে ফাইলের ফাংশনগুলো আরও অনেক বেশি অ্যাডভান্সড। যেমন সি শার্পে কোনো ফাইল হার্ডডিস্ক কোথায় আছে, তা বের করার জন্যই কয়েক ধরনের ফাংশন আছে। এগুলো সি-এর ফাংশনগুলোরই আরও অ্যাডভান্সড এডিশন কজ

ফিডব্যাক : wahid_cseaust@yahoo.com