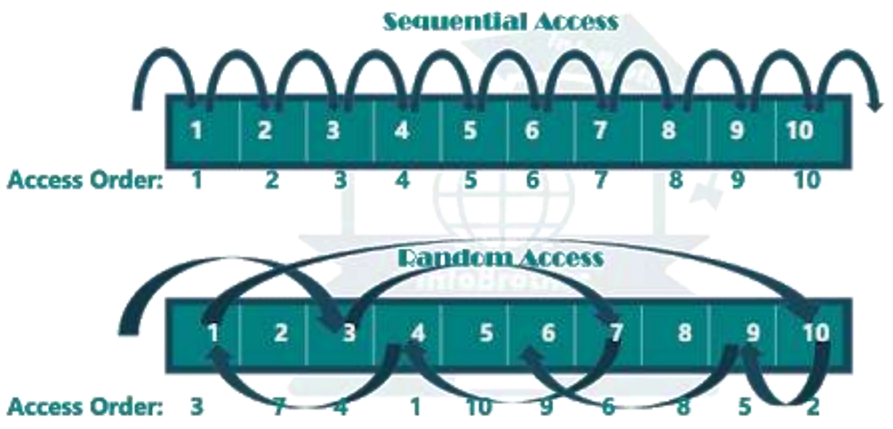
Random Access to files (File Handling)

An application can be created to access file information in a random fashion. In a sequential file, information is accessed in a chronological order whereas the random access provides instant fetching of a record and that too can in any order or no order at all. In random access, we must implement the capability to locate a record immediately. This is typical of any transaction system, be it banking, ticket reservation, and so forth



- Rather than reading all of the records until you get to the one you want, you can skip directly to the record you wish to retrieve.
- Random file access is done by manipulating the file pointer

What is File pointer (*fp)

Each file stream class contains a file pointer that is used to keep track of the current read/write position within the file.

When something is read from or written to a file, the reading/writing happens at the file pointer's current location.

By default, when opening a file for reading or writing, the file pointer is set to the beginning of the file

According to opening mode of file, file pointer can be categorized in two ways

- get pointer
- put pointer

1. get pointer

- ❖ The get pointer allows us to read the content of a file when we open the file in read-only mode.
- ❖ It automatically points at the beginning of file, allowing us to read the file from the beginning.

In order to perform a file input operation using the get pointer, C++ provides us a few file stream classes…

- ❖ Ifstream
- ❖ Fstream

File mode to work with get pointer?

ios::in = Searches for the file and opens it in read mode only and gives access to get pointer, to read the content of a file.

Functions to work with the get pointer?

- tellg()
- seekg()

tellg()

Gives us the current location of the get pointer. When the file is opened in a read-only mode, tellg() returns zero i.e. the beginning of the file.

Seekg()

This function is used to set the location of the get pointer to a desired position/offset. The position variable is the new position in the file i.e. an integer value representing the number of bytes from the beginning of the file.

Syntax:

1. seekg (streampos position); //one argument as offset
2. seekg (streamoff offset, ios_base::seekdir dir);//two argument

here, The position variable is relative to the dir parameter. dir is the seeking direction and it can take any of the following constant values:

ios::beg - offset from the beginning of the file.

ios::cur - offset from the current position in the file.

ios::end - offset from the end of the file.

Example 1

tellg()

```
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
   fstream file;

   // open file in read and write mode
   file.open("Hello.txt", ios::out);
   file << "Hello class";

   // print the position of the pointer in file
   cout << "the current position of pointer is :"
       << file.tellp() << endl;

   // close the open file
   file.close();
}
```

Example 2.

```
#include <iostream>
#include <fstream>
using namespace std;
int main ()
{
  fstream obj;
  obj.open ("test.txt", ios::in);
  char ch;
  int pos;
  while(!obj.eof())
  {
    obj>>ch;
    pos = obj.tellg();
    cout<<pos<<"."<<ch<<"\n";
  }
  obj.close();
}
```

Seekg()

```
#include <fstream>

#include <iostream>
using namespace std;
int main (  ) {
   fstream File("hello.txt", ios::in | ios::out );
   File << "Hello world";
   File.seekg(9, ios::beg);
```

```
  char F[9];
  File.read(F, 5);
  F[5] = 0;
  cout <<F<< endl;
  File.close();
}
```

Example 2:

```
#include<iostream>
#include<fstream>
using namespace std;
int main()
{

//Creating an input stream to read the content of a file
ifstream ifstream_ob;
//Opening a file named country1.txt to read its content

ifstream_ob.open("File1.txt", ios::in);
cout<<"The first location in the file : " <<ifstream_ob.tellg() <<
"\n";
char ch;
cout<<"\nReading the content of file : \n";
//Read the file until EOF is reached
while(ifstream_ob)
{
ch = ifstream_ob.get();
cout<<ch;
}

//Setting the EOF flag off, to allow the access of file again for
reading
ifstream_ob.clear();
cout<<"\n\nReading the content of file once again : \n";
//Taking the get pointer at the zero byte location from the
beginning of the file
ifstream_ob.seekg(0, ios::beg);

//Reading the content of the file again
while(ifstream_ob)
{
ch = ifstream_ob.get();
cout<<ch;
}
return 0;
}
```

2. Put Pointer

- ❖ It allows us to write the content to a file, when we open the file in write-only mode. i.e. ios::out.
- ❖ It automatically points at the beginning of a file, starting us to write the content of a file from the start.

File Output Stream Classes

- ❖ Ofstream
- ❖ Fstream

File modes to work with put pointer.

- ❖ Ios::out
- ❖ Ios::binary
- ❖ Ios::app
- ❖ Ios::ate etc

Functions to work with the put pointer

- ❖ tellp()
- ❖ seekp()

❖ tellp()

Gives us the current location of the put pointer. When the file is opened in a write-only mode, tellg() returns zero i.e. the beginning of the file.

Example:

```
#include <iostream>
#include <fstream>
using namespace std;

int main()
{
   fstream file;

   // open file in read and write mode
   file.open("Hello.txt", ios::out);
   cout<<file.tellp()<<endl;
   file << "Example of tellp";

   // print the position of the pointer in file
   cout << "the current position of pointer is :"
       << file.tellp() << endl;
       file<<"Random Access"<<endl;
       cout<<file.tellp()<<endl;

   // close the open file
   file.close();
}
```

❖Seekp()

This function is used to set the location of the put pointer to a desired position/offset. The position parameter is the new position in the file i.e. an integer value representing the number of bytes from the beginning of the file.

Syntax:

- ❖ seekp (streampos position);
- ❖ seekp (streamoff offset, ios_base::seekdir dir);

Example:1

```
#include <iostream>
#include <fstream>
using namespace std;
int main ()
{
  fstream obj;
  obj.open ("test.txt", ios::out);
  obj<<"Hello  World";
  int pos = 6;
  obj.seekp(pos-1);
  obj<<"...And here the text changed";
  obj.close();
  return 0;
}
```

Example 2;

```
#include<iostream>
#include <fstream>
int main () {
   std::ofstream outfile;
   outfile.open ("Hello.txt");
 outfile.write ("This is an apple",16);
   long pos = outfile.tellp();
   outfile.seekp (pos-7);
   outfile.write (" sai",4);
    outfile.close();
   return 0;
}
```

Example program of tellg(),tellp(),seekg(),skeep()

```
#include <iostream>

#include <fstream>
using namespace std;

int main()
{
   fstream F;
   // opening a file in input and output mode
   F.open("my.txt", ios::in | ios::out);

   // getting current location
   cout << F.tellg() << endl;

   // seeing 8 bytes/characters
   F.seekg(8, ios::beg);
   // now, getting the current location
   cout << F.tellg() << endl;
   // extracting one character from current location
   char c = F.get();
   // printing the character
   cout << c << endl;

   // after getting the character,
   // getting current location
   cout << F.tellg() << endl;
   // now, seeking 10 more bytes/characters
```

```
F.seekg(10, ios::cur);
// now, getting current location
cout << F.tellg() << endl;
// again, extracing the one character from current location
c = F.get();
// printing the character
cout << c << endl;

// after getting the character,
// getting current location
cout << F.tellg() << endl;
// again, seeking 7 bytes/characters from beginning
F.seekp(7, ios::beg);
// writting a character 'Z' at current location
F.put('Z');
// now, seeking back 7 bytes/characters from the end
F.seekg(-7, ios::end);
// now, printing the current location
cout << "End:" << F.tellg() << endl;
// extracting one character from current location
c = F.get();
// printing the character
cout << c << endl;

// closing the file
F.close();
return 0;
```