- Command Line Arguments (CLI)
- eof()
- Random access – seekp(), seekg()
-

(unit-5)

## Command line argument (CLI)

- CLI is a way to take i/p from the user.
- Using CLi i/p can be ~~fed when~~ provided (P|V) when user runs the prog on terminal
- For eg:- filename can be p|v as input when the user runs the prog on terminal

Q) How to p|v CLi?

A. When prog is run on the terminal the i/p is p|v after ./a.out in terminal

eg ./a.out file.txt

↓ Cmd to run the prog

↓ argument (i/p) by the user – filename.

Q) How is the argument (arg)/ input available in the prog?

A When the code is written, then main() fun will take 2 param.

1) arg count :-

int argc :- it will the No. of arguments p/v by the user.

- ./a.out → only 1 arg
- ./a.out file1.txt → 2 arg.
- ./a.out file1.txt file2.txt → 3 arg.

There shud be space b/w each arg.

No commas.

- This value is auto$^{y}$ supplied by OS

2) arg vector /array :-

char * argv[]

// The second param is a ~~th~~ array of char pointer.

// It will store the value of each arg.

eg ./a.out file1.txt file2.txt

↓ ↓ ↓

stored in argv[0] | stored in argv[1] | Stored in argv[2]

Note : when the prog is run, user must be aware to pass the arg in CLI o/w prog will not run correctly.

Eg Prog to read file name from CLI.

** //main fun shud have these param.

```
int main(int argc, char * argv[])
```

// I param will store arg count i.e # of arg.
// it can have any name but argc is generally used.
// ./a.out is also considered as arg.
// ./a.out file.txt
# of arg = 2.

// II param is an array of char pointer
// it will store value of each arg. Can have any name but argv is generally used.
// Values of arg will be.
argv[0] ⇒ ./a.out
argv[1] ⇒ file.txt

```
ifstream myin(argv[1]);
```
→ // filename will be in the 1st arg.

```
if (!myin)
{
    cout<<"Cant open";
    return 0;
}
char words[255];
myin.read(words, sizeof(word));
Cout<<words;
myin.clear();
}
```

// When this prog is run in terminal, it must be run as
./a.out file.txt

- If file name is not p/v then that arg will be considered as NULL
- So file will not open.

- If you r using any GUI/IDE then there r diff ways to suffly CLI.
- just google how to suffly CLI in <IDE>.

---

## eof() fun to detect end of file

- fstream p/v an eof() fun to detect end of file in a loop.

eg

```
Char ch;
while(1)
{
  myin>>ch
  if(myin) cout<<ch;
  else break;
// This condn will break loop.
// eof is detected when stream state is false
}
```

using eof

This loop can be written as this way.

```
while(!myin.eof())
{ //eof is detected here.
  myin>>ch;
  if(myin) cout>>ch;
// No need to specify loop break condn now
}
```

# Random Access in a file using seekp & seekg().

- All the fun that we have discussed earlier p/v sequential r/w to a file
- seekp() & seekg() allows random access.

seekp() → moves write/put ptr to a specific pos

seekg() → moves get/read ptr to a specific pos.

- stream maintains 2 ptr.

  getptr → determines where read opr will occur

  put ptr → " " write " "

seekp & seekg() moves there ptr to a specified pos.

**Note** :- • seekp() & seekg() don't r/w at that pos. They only move the ptr, after that you will have to use some other fun to r/w

## Using seekg() to move get ptr (read ptr)

Q) Begin reading from 8th pos in the file.
i.e 1st move the read ptr to 8th pos & then start reading from that pos.

A
```
int main()
{
    ifstream myin("file.txt");
    if(!myin)
    {
        cout<<"Cant open";
        return 0;
    }
    myin.seekg(8, ios::beg)
//move read ptr to 8th pos from
  begin of file
                ↙ 8th pos.
"This is Line1 \n
 This is Line2"

//Now start reading
 Char ch;
while(1)
{
    myin.get(ch);
    if(myin) cout<<ch;
    else break;

}
myin.close();
}
```

o/p will not be "This is line1
This is line2"

o/p will be "line1
This is line2".

---

Note seekg() will not read anything. It will simply specify the read pos.

---

## Using seekp() to move write ptr

Q) 1) Write "Happi new year" to a file.
2) Then change 'p' in 4th pos to 'y'.

A)
```
int main()
{
    ofstream myout("file1.txt");
    char text[] = "Happi new year";
    myout.write(text, sizeof(text));
    myout.close();
    // Happi new year is written to file
    myout.open("file1.txt", ios::in);
        // If u donot specify this mode, then
        all the data will be deleted.
```

```
myout.seekp(4, ios::beg);
//move the write pos to 4th pos from beg

//now change the char to 'y'
~~myout.seekp~~
myout.put('y');
  //'y' will be written to the specified
    pos.

myout.close();

}
```

o/p "The text "Haffie new year"
will become "Haffy new yea".