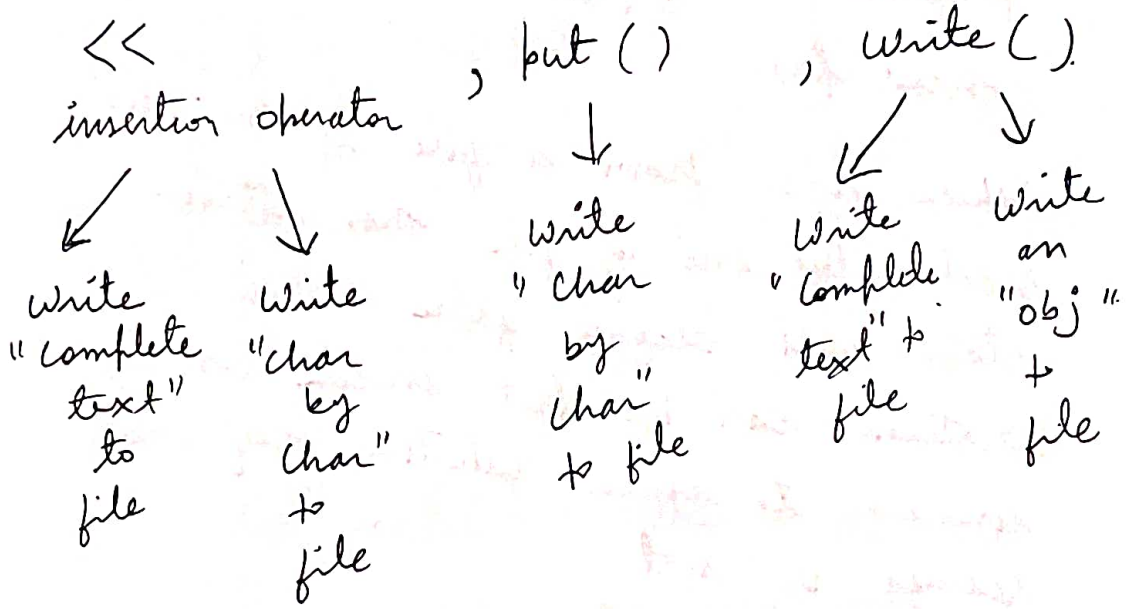
Ways to perform write operation to a file

(Unit-5)

- Writing to a file can be done using



Q How are current r/w position maintained in a file?

A Each stream will have 2 pointers
1) get ptr wh. tracks the current pos^n to read from a file
2) put ptr. wh. tracks the current pos to write to a file

- Whenever any read operation is performed then get ptr will move to next pos.
- Whenever, any r operation is performed then put ptr will move to next pos.
- While r/w from a file if the ptr reaches the end of file then stream state will become false.
- So stream can be put inside an if condition to check whether eof has reached or not.

eg
```
while(1)
{
    myin.get(ch);
    if(myin) cout>>ch;
    else break;
}
```

- Here myin stream is used to read from a file.
- get fun will read 1 char & move ptr to next pos.
- if(myin) condition will become false then loop will be broken.
- So this loop can be applied to read from a stream.

Writing to a file - 1

- Writing the "complete text" to a file using "<< operation"

- << can be used to write a block of text to a stream.
- No loop is needed.

Eg
```
#include <iostream>
#include <fstream> // include this hdr file to create streams

int main()
{
    char text[] = "This is L1\n This is L2";
        // L1, L2 means line1, line2.
    //This text needs to be written to the file

    step-1> Create a stream & open file to write
    ofstream myout("file.txt");

    step-2> Check if open success
    if (!myout)
    {
        cout << "Can't open file"; // Can't open. msg
        return 0; // Prog will exit
    }
```

step-3) write to a file

```
myout<<text;
```

//<< operator will write the complete text to this file.

step-4) close file:

```
myout.close();
```

}

Writing to a file-2

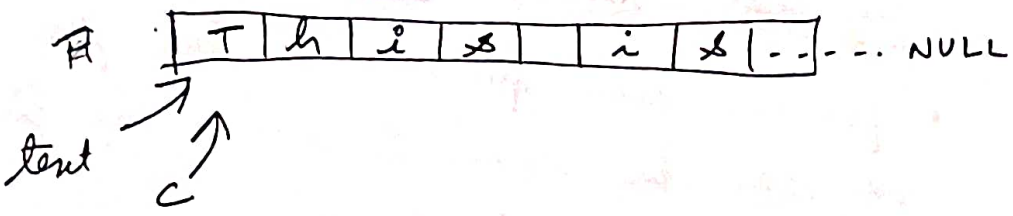
- Write the given text to a file "char by char" using "<< operator"
- Loop will be used.

eg

```
#include ...
....
....
int main()
{
    char text[] = "This is line1\n
                   This is line 2";

    S1) Create o/p stream
    ofstream myout("file.txt");
    S2) Check if open success
    if(!myout)
    {  cout<<"Can't open";
       return 0;
    }.
```

#3) use a pointer to access each char of array




```
char *c;
c = text; // don't write c = &text;
```

#4) use while loop to access each el of this array.

```
while(*c != NULL)
{
    myout << *c;
    //*c is used to access each char
    //write one char to file stream at a
      time

    c++;
    // increment pointer
}
```

#6) close stream

```
myout.close()
}
```

Writing to a file - 3

- Write the given text to a file "char by char" using "put" fun.
- fstream class provides a put fun wh. can be used to write one char at a time to a file.
- Syntax myout.put(ch);
- Put() fun takes one char as param wh. will be written to the file

Program:-

```
int main()
{
    char text[] = "This is L1 \n This is L2";
    ofstream myout("file.txt");
    if(!myout)
    {
        cout<<"Cant open";
        return 0;
    }
//use loop & ptr to access the text array char by char
char *c;
 c = text;
while(*c != NULL)
{
    myout.put(*c);
    c++;
}
```

//put fun is used to write 1 char
// to the file

myout.close()

}.

Writing to a file - 4

- ~~Write the given text to a file~~
- Write the "complete" given text to a file using "write fun"

- Write is a very convenient fun wh. is used to write the complete text to the file.

- syntax

```
myout.write(text, sizeof(text));
```

text → array to be written to file → must be of type char*.

sizeof(text) → size of arr.

- Write() takes 2 param
first: char array, must be of type char*
second: size of array.

```
int main()
{
    char text[] = "This is line 1\n This
                   is line 2";
    ofstream myout("file.txt");
    if(!myout)
    {
        cout<<"Cant open";
        return 0;
    }
    myout.write(text, sizeof(text));
    myout.close();
}
// No loop is reqd in this case.
```

Writing to a file-5

- Write a "class object" to a file using "write" fun()
- Write is very convenient & easy to use.
- Doesn't require any loop.
- It can write complete text to a file
- It can also be used to write an "obj" to a file

eg // Assume that student class is already created.

```
int main()
{
    student obj;
    obj.set(10,20,30); // assign values to var of this obj
    ofstream myout("file.txt");
    if (!myout)
    {
        cout<<"cant open";
        return 0;
    }
```

```
myout.write((char*)&obj, sizeof(obj));
```

first param shud be a pointer. obtain a ptr to obj & convert to char*

second param is size of obj.

```
    myout.close();
}
```

Summary of different ways to write to a file

Method	description	Use	while loop required
<< (insertion) operation	write complete text	myout<<text;	No
	write char by char	mbyout<<ch;	Yes.
put()	write char by char.	myout.put(ch);	Yes.
write fun	write complete text	myout.write (text, sizeof(text));	No.
	write an obj	myout.write ((char*)&obj, sizeof(obj))	No.

Note • Using write fun is the most convenient way to write a text to the file.

• But sometimes you may need to process the text char by char. So use use put() to write char by char to a file.