# Level 2: Reading / Writing Files

Look at the following example:

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

int main ()
{
  int age;
  cout << "Hi, how old are you?";
  cin >> age;
  cout << "You are " << age << " years old.";
}
```

You will see on the screen as follows:

```
Hi, how old are you?
14
You are 14 years old.
```

`cout` and `cin` commands are used to interact with the user. This way, we write sentences to the screen to ask questions or to say something and get answers from the user. What we write to the screen is called '*output*' and what we are reading from the user is '*input*'. Computer programs get the input, process it, and print the output. Sometimes, reading from the keyboard and writing to the screen is not practical; every time the program runs we have to enter from keyboard. Instead, we can use files; we can write our input once into the files and the program reads from the file each time. Moreover, we can save the output in another file for further use instead of writing to screen.

```
1.   #include <fstream>
2.   using namespace std;
3.
4.   int main ()
5.   {
6.     int age;
7.     ifstream fin ("inFile.txt");
8.     ofstream fout ("outFile.txt");
9.     fin >> age;
10.    fout << "You are " << age << " years old.";
11.  }
```

| *inFile.txt*: | *outFile.txt*: |
|---|---|
| `14` | `You are 14 years old.` |

Notice that line 1 is different than the first line of the other program. We have to write this line to read from a file and write to another one. At line 7, we say that we want to read our input from *inFile.txt* file. Similarly at line 8, we say that we want to write our output to *outFile.txt*. Other file names could also be used. At line 9, we use `fin` instead of `cin` to read the input from the file instead of the keyboard. At line 10, we use `fout` instead of `cout` to write the output to the file instead of the screen.

Before running the program we open *inFile.txt* and write the required input in it. In the example above, the input is 14. Since we assume that the input is already in the file when we run the program, we do not need to inform the user to enter the input as we did in the first example.

As with `cin`, we can read multiple values at once with `fin` as shown in the example below:

```
1.   #include <fstream>
2.   using namespace std;
3.
4.   int main ()
5.   {
6.     int num1, num2, num3;
7.     ifstream fin ("inFile.txt");
8.     ofstream fout ("outFile.txt");
9.     fin >> num1 >> num2 >> num3;
10.    fout << "The sum of these three numbers is "
11.         << num1 + num2 + num3 << ".";
12.  }
```

| *inFile.txt*: | *outFile.txt*: |
|---|---|
| `5 10`<br>`12` | `The sum of these three numbers is 27.` |

At line 9, we want to read three numbers for our three boxes. The numbers go to the boxes in the same order; 5 goes to `num1`, 10 goes to `num2`, and 12 goes to `num3`. Note that three numbers are not at the same line in the file, and it does not really matter. When `fin` asks for a number, the first following number will be read, no matter where it is.

*Example*: Write a program that reads a number N, and after that N numbers from a file. The program will write the sum of these N numbers to the output file.

| *inFile.txt*: | *outFile.txt*: |
|---|---|
| 5<br>1 2 3 4 6 | The sum is 16. |

*Solution*:

```
1.   #include <fstream>
2.   using namespace std;
3.
4.   int main ()
5.   {
6.     int N;
7.     int numbers[100];
8.     int sum = 0;
9.     ifstream fin ("inFile.txt");
10.    ofstream fout ("outFile.txt");
11.    fin >> N;
12.    for (int counter = 0; counter < N; counter++)
13.      fin >> numbers[counter];
14.    for (int counter = 0; counter < N; counter++)
15.      sum += numbers[counter];
16.    fout << "The sum is " << sum << ".";
17.  }
```

First we read `N` from the file at line 11. Then, in a `for` loop, we read and put each number in an array by using `counter` from 0 to N-1 at lines 12-13. Next, we add the numbers up in another `for` loop which traverses the array at lines 14-15. Finally at line 16, we write the answer to the file.

In the example above, N is given in the beginning. Suppose N is not given and we have to read the numbers till the end of the file.

| *inFile.txt*: | *outFile.txt*: |
|---|---|
| 1 2 3 4 6 | The sum is 16. |

We can write the program as follows:

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

int main ()
{
```

```
  int N;
  int numbers[100];
  int sum = 0;
  ifstream fin ("inFile.txt");
  ofstream fout ("outFile.txt");
  for (N = 0; ; N++)
  {
    fin >> numbers[N];
    if (fin.eof()) break;
  }
  for (int counter = 0; counter < N; counter++)
    sum += numbers[counter];
  fout << "The sum is " << sum << ".";
}
```

Here, we determine N by ourselves by incrementing each time reading a number from the input file. We assume the input file has an extra line after the last number. In the last turn before the break in the loop, `fin` reads that extra line, a number cannot be read from the input file, and input file ends – `fin.eof()` becomes true. Be careful that if you write the `fin.eof()` in for loop's condition, N will be incremented one more.

*Exercise 1*: Solve the same problem without using arrays.