Reading from files

When we read (extract) data or values from a file, we can use the file extractor operator >> in the same way as we do when we read data from the keyboard.

However, we have to keep in mind that there is a limited number of data values in a file, **but** we do not know how many values there are in the file. Therefore we typically use a `while` loop that test for the end of the file when we process a file.

We can test for the end of a file in two ways:

1. extracting or reading values from the file until the file is empty (see pages 266 - 369 in Savitch 10th edition), or
2. using the `eof()` function (see pages 387 - 388 in Savitch 10th edition).

Example 1:
Suppose we have a file called `myfile.dat` with the following names and ages of people:

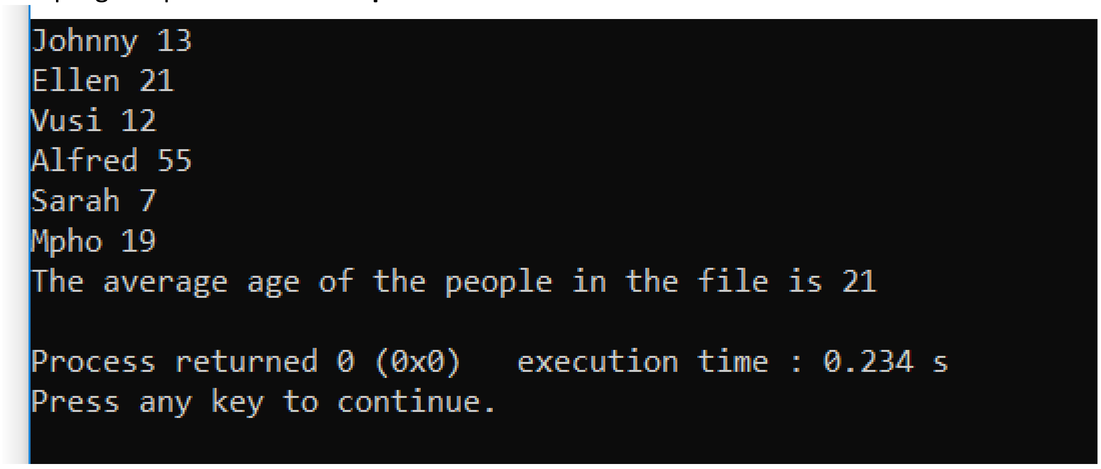
```
Johnny 13
Ellen 21
Vusi 12
Alfred 55
Sarah 7
Mpho 19
```

The following program is a simple program to read the names and corresponding ages, display it on the console and calculate the average age:

```
#include <iostream>
#include <fstream>
#include <cstdlib>

using namespace std;

int main()
{
    ifstream infile;

    infile.open("myfile.dat");
    if (infile.fail())
    {
        cout << "Input file opening file myfile.dat failed";
        exit (1);
    }

    string name;
    int age, averageAge = 0, sum = 0, nrPeople = 0;
    while (infile >> name >> age)
    {
        nrPeople++;
        cout << name << " " <<age << endl;
        sum += age;
    }
    averageAge = sum/nrPeople;
    cout << "The average age of the people in the file is "
```

```
            << averageAge << endl;

    return 0;
}
```

The program produces this **output**:



```
Johnny 13
Ellen 21
Vusi 12
Alfred 55
Sarah 7
Mpho 19
The average age of the people in the file is 21

Process returned 0 (0x0)   execution time : 0.234 s
Press any key to continue.
```

The statement

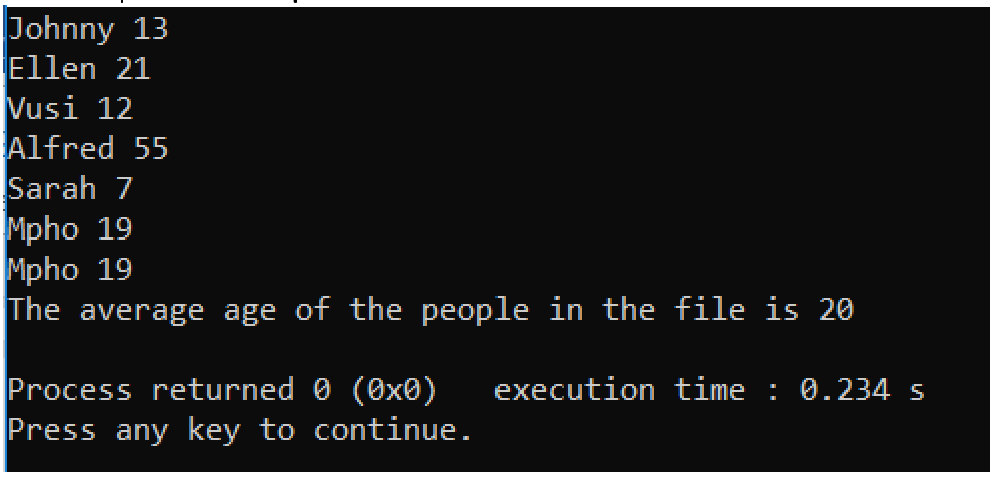
```
    while (infile >> name >> age)
```

both reads the name and age of a person from the file and checks for the end of file at the same time. As soon as the last name and age have been read form the file, the end-of-file (eof) member function will become true and the loop will be terminated.

Example 2:
If we try to achieve the same effect by using the `eof()` function in the while loop to test that we have not yet reached the end of the file as shown below:

```
#include <iostream>
#include <fstream>
#include <cstdlib>

using namespace std;

int main()
{
    ifstream infile;

    infile.open("myfile.dat");
    if (infile.fail())
    {
        cout << "Input file opening file myfile.dat failed";
        exit (1);
    }

    string name;
    int age, averageAge = 0, sum = 0, nrPeople = 0;
    while (!infile.eof())
    {
        infile >> name >> age;
        nrPeople++;
        cout << name << " " <<age << endl;
        sum += age;
    }
    averageAge = sum/nrPeople;
    cout << "The average age of the people in the file is "
         << averageAge << endl;

    return 0;
}
```

We end up with this **output**:



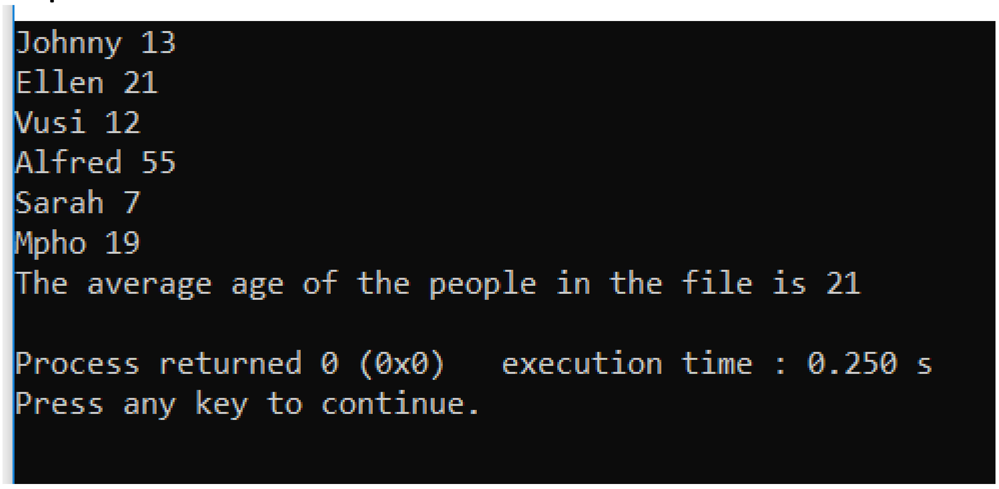
```
Johnny 13
Ellen 21
Vusi 12
Alfred 55
Sarah 7
Mpho 19
Mpho 19
The average age of the people in the file is 20

Process returned 0 (0x0)   execution time : 0.234 s
Press any key to continue.
```

NB: Note that the last person and his age (Mpho 19) are displayed twice. The reason for this is because the `eof` file function only becomes true, once the program has read ***past*** the end of the file. To ensure our program executes correctly, we have to adapt the code as follows:

Example 3:

```
#include <iostream>
#include <fstream>
#include <cstdlib>

using namespace std;

int main()
{
    ifstream infile;

    infile.open("myfile.dat");
    if (infile.fail())
    {
        cout << "Input file opening file myfile.dat failed";
        exit (1);
    }

    string name;
    int age, averageAge = 0, sum = 0, nrPeople = 0;
    infile >> name >> age;
    while (!infile.eof())
    {
        nrPeople++;
        cout << name << " " <<age << endl;
        sum += age;
        infile >> name >> age;
    }
    averageAge = sum/nrPeople;
    cout << "The average age of the people in the file is "
         << averageAge << endl;

    return 0;
}
```

Output:



```
Johnny 13
Ellen 21
Vusi 12
Alfred 55
Sarah 7
Mpho 19
The average age of the people in the file is 21

Process returned 0 (0x0)   execution time : 0.250 s
Press any key to continue.
```

We typically use the code as shown in Example 1 to read values from a data file.

We can also read an object from a data file in the same way, if we overload the stream extraction operator >> to do so.

Suppose we have a class `Person` defined as follows. We overload the stream extraction operator >>:

Person.h:

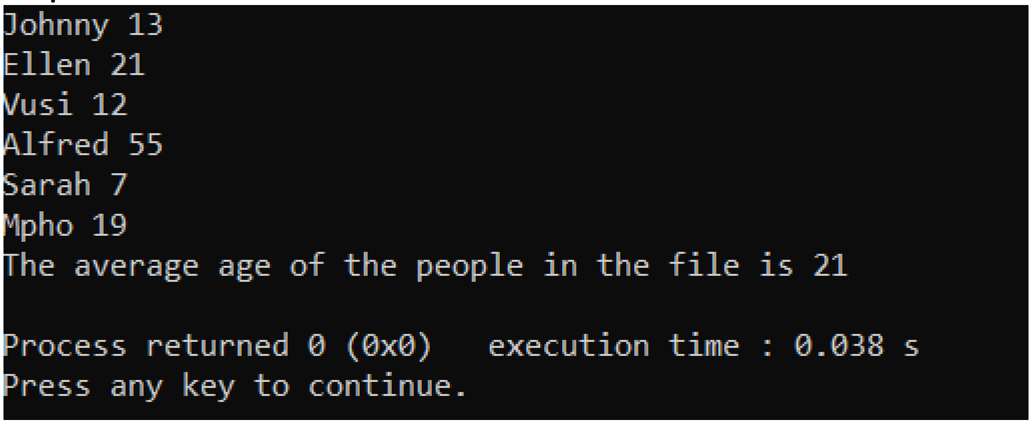
```
#ifndef Person_h
#define Person_h
#include <iostream>
#include <fstream>
#include <cstdlib>

using namespace std;

class Person
{
    public:
        friend istream& operator >> (istream & inf, Person & p);
        Person();
        ~Person();
        string getName();
        int getAge();
    private:
        string name;
        int age;
};
#endif
```

Person.cpp
```
#include "Person.h"
#include <fstream>
#include <iostream>
#include <cstdlib>

using namespace std;

istream& operator >> (istream & inf, Person & p)
{
    inf >> p.name >> p.age;
    return inf;
}
Person::Person()
{
    name ="";
    age = 0;
}

Person::~Person()
{

}

string Person::getName()
{
    return name;
}

int Person::getAge()
{
    return age;
}
```

We can then use the overloaded operator >> to read the data for one person from our file `myfile.dat` as shown in the main.cpp application file below. Also note how we now use the accessor functions `getName()` and `getAge()` to obtain the values of the member variables `name` and `age` in the `p` object (of class `Person`) :

Main.cpp:
```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include "Person.h"

using namespace std;

int main()
{
    ifstream infile;

    infile.open("myfile.dat");
    if (infile.fail())
    {
        cout << "Input file opening file myfile.dat failed";
        exit (1);
    }

    Person p;
    int  averageAge = 0, sum = 0, nrPeople = 0;
    while (infile >> p)
    {
        nrPeople++;
        cout << p.getName() << " " <<p.getAge() << endl;
        sum += p.getAge();
    }
    averageAge = sum/nrPeople;
    cout << "The average age of the people in the file is "
         << averageAge << endl;

    return 0;
}
```

Output:



```
Johnny 13
Ellen 21
Vusi 12
Alfred 55
Sarah 7
Mpho 19
The average age of the people in the file is 21

Process returned 0 (0x0)   execution time : 0.038 s
Press any key to continue.
```

We usually use a `while` loop where we read the data directly from the file, when we work with data files, and the `eof` function when we work with text files.

NB Study the notes on differences between a data file and a text file also.