Introduction to Programing and Matlab

Dr. Hyun Lim
h.lim@lboro.ac.uk

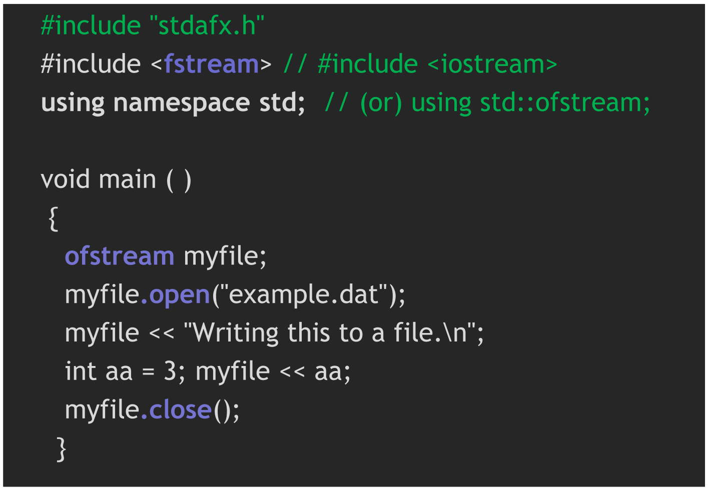
Institute for Digital Technologies
Loughborough University London

C Programming

- File Input/output

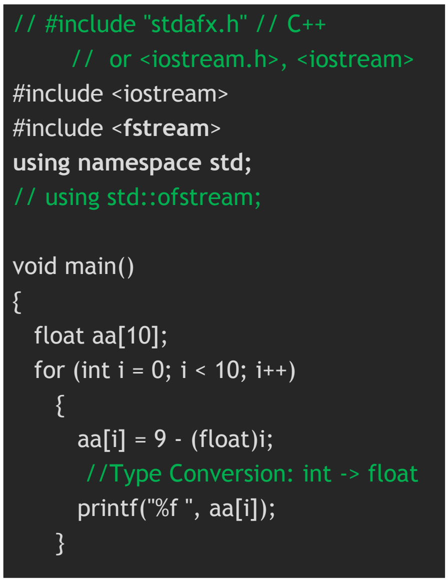
Input to a file in C++

- **C++** provides the **fstream** **class** to both read and write from/to text files (e.g. *.txt*, *.dat*).

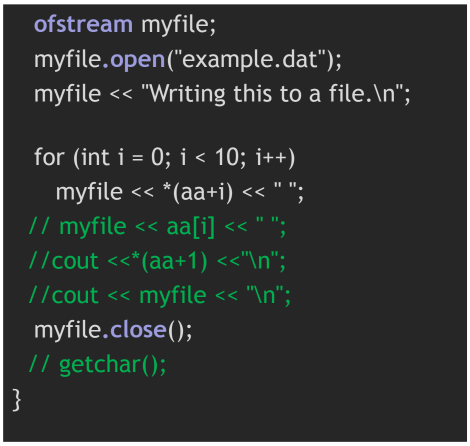


```
#include "stdafx.h"
#include <fstream> // #include <iostream>
using namespace std;  // (or) using std::ofstream;

void main ( )
 {
   ofstream myfile;
   myfile.open("example.dat");
   myfile << "Writing this to a file.\n";
   int aa = 3; myfile << aa;
   myfile.close();
  }
```

Input to a file in C++



```
// #include "stdafx.h" // C++
      //  or <iostream.h>, <iostream>
#include <iostream>
#include <fstream>
using namespace std;
// using std::ofstream;

void main()
{
  float aa[10];
  for (int i = 0; i < 10; i++)
    {
      aa[i] = 9 - (float)i;
       //Type Conversion: int -> float
      printf("%f ", aa[i]);
    }
```



```
  ofstream myfile;
  myfile.open("example.dat");
  myfile << "Writing this to a file.\n";

  for (int i = 0; i < 10; i++)
    myfile << *(aa+i) << " ";
 // myfile << aa[i] << " ";
 //cout <<*(aa+1) <<"\n";
 //cout << myfile << "\n";
  myfile.close();
 // getchar();
}
```

Input to a file in C

```
// #include "stdafx.h" // C++
       //  or <iostream.h>, <iostream>
 #include "stdio.h"   // C  or <stdio.h>
 #include <conio.h> // C kbhit()

void main()
 {
   float aa[10];
   for (int i = 0; i < 10; i++)
     {
       aa[i] = 9 - (float) i;
              // because int i, float aa[]
       printf("%f ", aa[i]);
     }
    // printf(" \n ");
```

```
 FILE *file_ptr;
 file_ptr = fopen("my_data.dat", "w");

 for (int j = 0; j < 10; j++)
    fprintf(file_ptr, "%f ", aa[j]);

 fclose(file_ptr);
  // The fclose closes the file.
 while(!kbhit());
}
```

fopen opens a file, e.g. *my_data.dat*, in the write (**w**) mode. If the file does not exist it will be created. But, be careful! If the file exists, the previous file will be destroyed and a new file is created instead. **fopen** returns a pointer to the file, which is stored in the variable *file_ptr*.

Output from a file in C - *fgetc()*

```
// #include "stdafx.h" // C++
 #include "stdio.h" // C  or <stdio.h>
 #include <conio.h> // C kbhit()

 void main() {
   float aa[10];
   for (int i = 0; i < 10; i++)
     {
        aa[i] = 9 - (float) i;
        printf("%f ",aa[i]);
     }
    // printf(" \n ");

  FILE * file_ptr;
  file_ptr = fopen("my_data.dat", "w");

  for (int j = 0; j < 10; j++)
     fprintf(file_ptr, "%f ", aa[j]);

  fclose(file_ptr);
```

```
 printf("\n Read the file:\n");

 #define PATH "C:\\......\\my_data.dat "
 FILE *f;
 char c;
 f = fopen("my_data.dat", "rt");
 // f = fopen(PATH, "rt");
 while ((c = fgetc(f)) != EOF)
    printf("%c", c);
 fclose(f);

 while(!kbhit());
}
```

You have to open a (text) file for reading in the read (**r**) mode. Then **fgetc** can be used to read the contents of the file. **fgetc** reads one character at a time from the file pointed by the FILE pointer

fgets()

- **fgets()** will read the **whole string** up to the size specified in argument list but when end of line occurs **fgetc()** returns **EOF** while **fgets()** returns **NULL**.

"input.txt" is opened for reading using the function fopen in the mode read (**r**). The library function **fgets** will read each line (with a maximum of 1000 characters per line.) If the end-of-file (**EOF**) is reached the fgets function will return a **NULL** value. Each line will be printed on stdout (normally your screen) until the EOF is reached.

```
 #include<stdio.h>

void main()
 {
   FILE *ptr_file;
   char buf[1000];

   ptr_file =fopen("my_input.txt","r");

   while (fgets(buf,1000, ptr_file)!=NULL)
       printf("%s",buf);

  fclose(ptr_file);
}
```

File I/O (read/write) modes

r - open for reading

w - open for writing

a - open for appending

r+ - open for reading and writing, start at the beginning

w+ - open for reading and writing (**overwrite** file)

a+ - open for reading and writing (**append** if file exists)

- ✓ **t**: Open in text (**translated**) mode.
- ✓ **b**: Open in **binary** (**untranslated**) mode
- ✓ If t or b is not given in mode, the default **translation mode** is defined.