

PROGRAMMING LANGUAGES 1

Input & Output



AREA OF RECTANGLE

```
#include <iostream>

using namespace std;

int main()
{
    float width, length, area; // Define variables needed by program

    cout << "Enter width: ";          // Ask user to enter width
    cin >> width;                      // Read width from keyboard

    cout << "Enter length: "; // Ask user to enter length
    cin >> length;                  // Read length from keyboard

    area = width * length;           // Calculate area

    // Now, print the calculated area

    cout << "Area of rectangle of width " << width << " and length " << length << " is " << area << endl;

    return 0;
}
```

cout

- Writes output to standard output
- Standard output is screen by default
- Can be directed to other output, e.g. file
- Followed by << operator
- Many outputs can be chained into single cout statement
- Examples:
 - `cout << "Hello World!";`
 - `cout << "Area of rectangle of width " << width
<< " and length " << length << " is " << area << endl;`

endl

- Short for “**end** line”
- Similar to “\n”
- Moves next output to new line

ESCAPE SEQUENCES

- `\n` → New line
- `\t` → Tab
- `\'` → Single quote `'`
- `\"` → Double quote `"`
- `\\` → Back slash `\`

cin

- Reads input from standard input
- Standard input is keyboard by default
- Followed by >> operator
- Input is read by the program when the user presses a whitespace key
 - Enter
 - Tab
 - Space
- Two or more inputs can be chained into single cin statement
- Examples :
 - `cin >> width;`
 - `cin >> width >> length;`

FILE INPUT & OUTPUT

- ifstream: input file stream
- ofstream: output file stream
- fstream: Input & output file stream

FILE INPUT & OUTPUT - EXAMPLE

```
#include<fstream>

using namespace std;

int main(){
    fstream file;
    file.open("file", ios::in | ios::out);    // open file for reading and writing
    ...
    file >> s;    // Read from file into s
    file << "Hello";    // Write to file
    ...
    file.close();
}
```

FILE MODE FLAGS

- `ios::in`
- `ios::out`
- `ios::app`
- `ios::binary`

FILE STREAM FUNCTIONS

- **open()**: open file
- **is_open()** : check if file is open
- **getline()** : get line
- **eof()** : true if reached end of file while reading
- **close()** : close file