Programming in C++

Programs with IO & Loop

Program: Hello World

C Program	C++ Program

// FileName:HelloWorld.c: #include <stdio.h>
 	
int main() {
 	
printf("Hello World in C"); printf("\n");
 	
return 0;
}
	// FileName:HelloWorld.cpp: #include <iostream>
	
	int main() {
	
	std::cout << "Hello World in C++"; std::cout << std::endl;
	
	return 0;
	}

Hello World in C	Hello World in C++
• IO Header is stdio.h • printf to print to console • Console is stdout file • printf is a variadic function • \n to go to the new line • \n is escaped newline character	• IO Header is iostream • Operator << to stream to console • Console is std::cout ostream (in std namespace) • Operator << is a binary operator • std::endl (in std namespace) to go to the new line • std::endl is stream manipulator (newline) functor

Program: Add two numbers

C Program

```
// FileName:Add_Num.c:
#include <stdio.h>
int main() {
        int a, b; int
        sum;
        printf("Input two numbers:\n");
        scanf("%d%d", &a, &b);
        sum = a + b;
        printf("Sum of %d and %d", a, b); printf("
        is: %d\n", sum);
        return 0;
}
```

```
Input two numbers:
3 4
Sum of 3 and 4 is: 7
```

- scanf to *scan* (*read*) from console
- Console is stdin file
- scanf is a variadic function
- Addresses of a and b needed in scanf
- All variables a, b & sum declared first (C89)
- Formatting (%d) needed for variables

C++ Program

```
// FileName:Add_Num_c++.cpp:
#include <iostream>
int main() {
        int a, b;

        std::cout << "Input two numbers:\n"; std::cin
        >> a >> b;

        int sum = a + b; // Declaration of sum

            std::cout << "Sum of "
                  << a << " and "
                   << b << " is: "
                   << sum << std::endl;

        return 0;
}
```

```
Input two numbers:
3 4
Sum of 3 and 4 is: 7
```

- operator>> to *stream* from console
- Console is std::cin istream (in std namespace)
- operator>> is a binary operator
- a and b can be directly used in operator>> operator
- sum may be declared when needed
- Formatting is derived from type (int) of variables

Program: Square Root of a number

C Program

```
// FileName:Sqrt.c: #include
<stdio.h> #include <math.h>


int main() {
        double x; double sqrt_x;


        printf("Input number:\n"); scanf("%lf",
        &x);

        sqrt_x =
                sqrt(x);

        printf("Sq. Root of %lf is:", x); printf(" %lf\n",
        sqrt_x);

        return 0;
}
```

```
Input number:
2
Square Root of 2.000000 is: 1.414214
```

- Math Header is math.h (C Standard Library)
- Formatting (%lf) needed for variables
- sqrt function from C Standard Library
- Default precision in print is 6

C++ Program

```
// FileName:Sqrt_c++.cpp: #include
<iostream> #include <cmath>
using namespace std;


                int main() {
                    double x;


        cout << "Input number:" << endl; cin >> x;


        double sqrt_x =         // Declaration of sqrt_x sqrt(x);


        cout << "Sq. Root of " << x;
        cout << " is: " << sqrt_x << endl;

        return 0;
}
```

```
Input number:
2
Square Root of 2 is: 1.41421
```

- Math Header is cmath (C Standard Library in C++)
- Formatting is derived from type (double) of variables
- sqrt function from C Standard Library
- Default precision in print is 5 (different)

namespace std for C++ Standard Library

C Standard Library	C++ Standard Library
• All names are global • stdout, stdin, printf, scanf	• All names are within std namespace • std::cout, std::cin • Use using namespace std; to get rid of writing std:: for every standard library name

W/o using

```
#include <iostream>

int main() {

        std::cout << "Hello World in C++"
                  << std::endl;

    return 0;
}
```

W/ using

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

    int main() {

    cout << "Hello World in C++"
         << endl;

}       return 0;
```

Standard Library Header Conventions

	C Header	C++ Header
C Program	Use .h. Example: #include <stdio.h> *Names in global namespace*	Not applicable
C++ Program	Prefix c, no .h. Example: #include <cstdio> *Names in std namespace*	No .h. Example: #include <iostream>

Any C standard library header is to be used in C++ with a prefix 'c' and without the .h. These symbols will be in std namespace. Like:

```
#include <cmath>  // In C it is <math.h>
...
std::sqrt(5.0);  // Use with std::
```

It is possible that a C++ program include a C header as in C. Like:

```
#include <math.h> // Not in std namespace
...
sqrt(5.0);   // Use without std::
```

This, however, is not preferred.

Using .h with C++ header files, like iostream.h, is disastrous. These are deprecated. It is dangerous, yet true, that some compilers do not error out on such use. Exercise caution.

Program: Sum n natural numbers

C Program

```
// FileName:Sum_n.c:
#include <stdio.h>


int main() { int n; int i;
        int sum = 0;

        printf("Input limit:\n"); scanf("%d", &n);

        for (i = 0; i <= n; ++i) sum = sum + i;

        printf("Sum of %d", n);
        printf(" numbers is: %d\n", sum);

        return 0;
}
```

```
Input limit:
10
Sum of 10 numbers is: 55
```

- i must be declared at the beginning (C89)

C++ Program

```
// FileName:Sum_n_c++.cpp: #include
<iostream>
using namespace std;

int main() { int n;

        int sum = 0;

        cout << "Input limit:" << endl; cin >> n;

        for (int i = 0; i <= n; ++i) // Local Decl.
        sum = sum + i;

        cout << "Sum of " << n ;
        cout << " numbers is: " << sum << endl;

        return 0;
}
```

```
Input limit:
10
Sum of 10 numbers is: 55
```

- i declared locally in for loop

Program: Using bool

	C Program	C++ Program

// FileName:bool.c: #include <stdio.h> #define TRUE 1 #define FALSE 0
 	
int main() {
int x = TRUE;
printf("bool is %d\n", x);
 	
return 0;
}
	// FileName:bool.c: #include <stdio.h> #include <stdbool.h>
	
	int main() {
	bool x = true;
	printf("bool is %d\n", x);
	
	return 0;
	}
		// FileName:bool_c++.cpp: #include <iostream>
		using namespace std; int
		
		main() {
		bool x = true;
		cout <<"bool is " << x;
		return 0;
		}

bool is 1	bool is 1	bool is 1
• Using int and #define for bool • May use Bool (C99)	• stdbool.h included for bool • Bool type & macros (C99): bool which expands to Bool • true which expands to 1 false which expands to 0	• No additional headers required • bool is a built-in type true is a literal false is a literal