1 Functions as arguments

Functions with functions as arguments are called *higher order* functions.

1.1 Sums

For example, the sum of an arithmetic progression is:

$$1 + 2 + 3 + \cdots + n = \sum_{i=1}^{n} i = \frac{n \times (n+1)}{2}$$

1.2 SML

Implemented in SML:

```
SML
(* sum : int → int → (int → real) → real *)
fun sum i n f = if i > n
                then 0.0
                 else f i + sum (i+1) n f ;
```

The function sum takes a beginning integer value of i, and ending integer value of n, and a function f that takes an integer and produces a real value. Since the term in this progression is just i, we just need function f to be the *identify* function, but to keep the types consistent, we will create a function to convert integer to real.

```
SML
fun int2real x = real x ;
```

With these two functions defined, we can test our code:

```
SML
sum 1 10 int2real ;  (* expect 55.0 *)
```

1.3 C++

Translating these functions to C++:

C++ source code written to file higherOrder.cpp

```
#include <iostream>
double sum(int i, int n, double (*f)(int))
{
  if(i > n) return 0.0;
  else return f(i) + sum(i+1,n,f);
}
double int2real(int x) { return (double) x; }
int main(){ std::cout << sum(1,10,int2real) << std::endl; }
```

Compiling and running the C++ code returns the same answer: 55.

1.4 Testing

Here are some shell scripts to make it simpler to build and run these files. But first run **docsml higherOrder.doc** to extract them!

Text written to file labdoc.sh

```
doctex higherOrder.doc && texdvi higherOrder.tex && dvips higherOrder.dvi
gsview higherOrder.ps &
```

Text written to file labcpp.sh

```
g++ -o higherOrder higherOrder.cpp
./higherOrder
```

Bourne Shell

```
chmod 755 labdoc.sh && chmod 755 labcpp.sh
```