MA122 - Computer Programming and Apllications

Indian Institute of Space Science and Technology

March 03, 2017

Lecture 18

1 Sorting

2 Inline function

3 Reference Variable

example


```
1  #include<iostream>
2  bool EvensFirst(int nX, int nY);
3  void SelectionSort(int *anArray, int nSize, bool (*
       pComparison)(int, int));
4  bool Ascending(int nX, int nY);
5  bool Descending(int nX, int nY);
6  void PrintArray(int *pArray, int nSize);
7  using namespace std;
8  int main()
9  {
10     using namespace std;
11
12     int anArray[9] = { 4, 7, 9, 5, 6, 1, 8, 2, 3 };
13
14     SelectionSort(anArray, 9, EvensFirst);
15     PrintArray(anArray, 9);
16     return 0;
17 }
```

example


```
1  bool EvensFirst(int nX, int nY)
2  {
3      // if nX is not even and nY is, nY goes first
4      if ((nX % 2) && !(nY % 2))
5          return false;
6
7      // if nX is even and nY is not, nX goes first
8      if (!(nX % 2) && (nY % 2))
9          return true;
10
11     // otherwise sort in Ascending order
12     return Ascending(nX, nY);
13 }
```

example


```
1  void SelectionSort(int *anArray, int nSize, bool (*
       pComparison)(int, int))
2  {
3  for (int nStartIndex= 0; nStartIndex < nSize;
       nStartIndex++)
4  {
5    int nBestIndex = nStartIndex;
6
7    for (int nCurrentIndex = nStartIndex + 1;
         nCurrentIndex < nSize; nCurrentIndex++)
8    {
9      if (pComparison(anArray[nCurrentIndex], anArray[
           nBestIndex]))
10                 nBestIndex = nCurrentIndex;
11   }
12   swap(anArray[nStartIndex], anArray[nBestIndex]);
13 }
14 }
```

example


```
1  bool Ascending(int nX, int nY)
2  {
3      return nY > nX;
4  }
5
6  bool Descending(int nX, int nY)
7  {
8      return nY < nX;
9  }
10
11 void PrintArray(int *pArray, int nSize)
12 {
13     for (int iii=0; iii < nSize; iii++)
14         cout << pArray[iii] << " ";
15     cout << endl;
16 }
```

Lecture 18

1 Sorting

2 Inline function

3 Reference Variable

example


```
1  #include <iostream>
2  #define SQUARE(X) X*X
3  // an inline function definition
4  inline double square(double x) { return x * x; }
5
6  int main()
7  {
8    using namespace std;
9    double a, b;
10   double c = 13.0;
11   a = square(5.0);
12   b = square(4.5 + 7.5); // can pass expressions
13
14   cout << "a = " << a << ", b = " << b << "\n";
15   cout << "c = " << c;
16   cout << ", c squared = " << square(c++) << "\n";
17   cout << "Now c = " << c << "\n";
```

example


```
1     int e=13.0;
2
3     cout<<SQUARE(4.5 + 7.5)<<endl;
4     cout<<SQUARE(e++)<<endl;
5     cout<<e<<endl;
6
7   return 0;
8 }
```

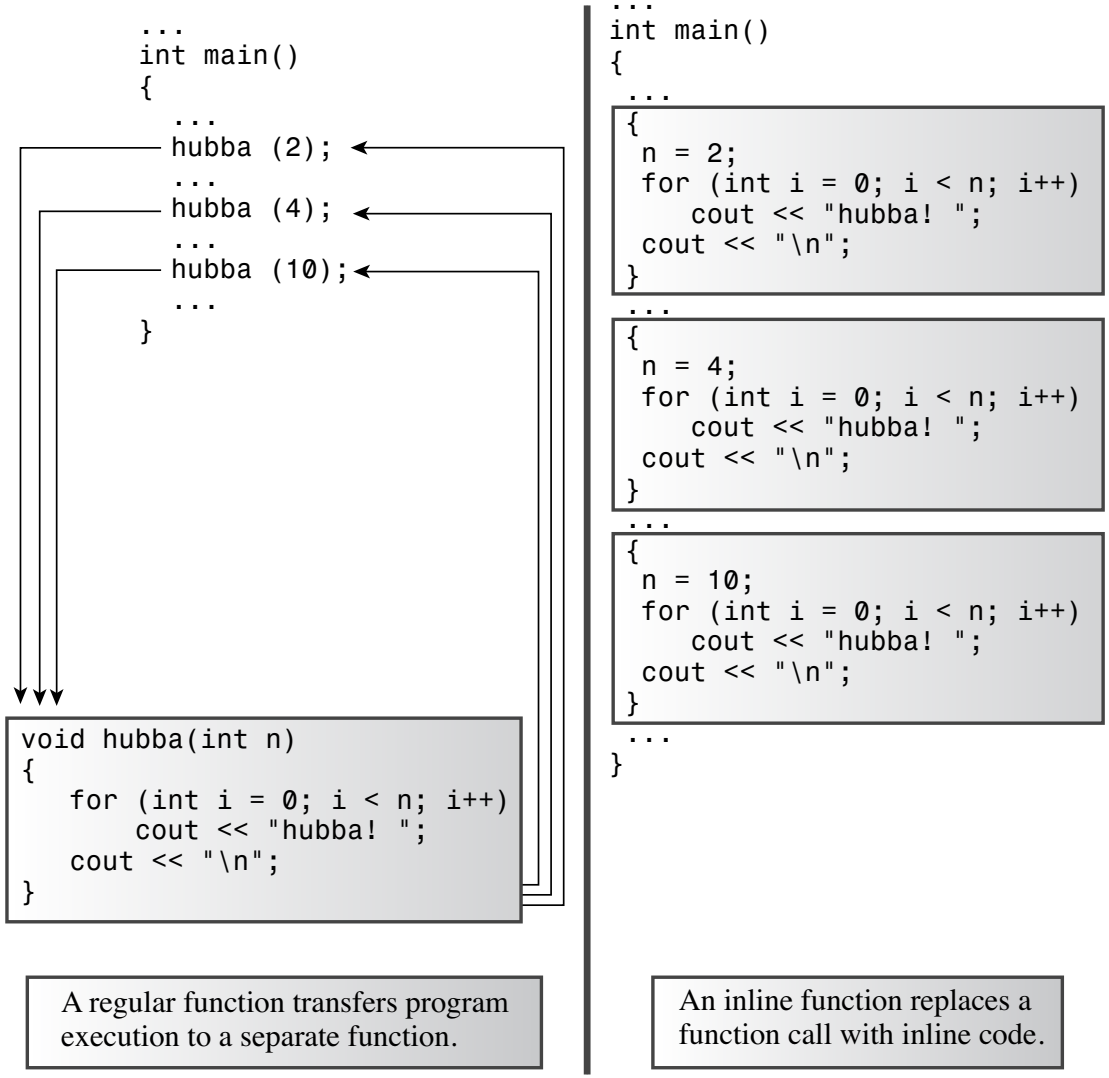

Figure 8.1 Inline functions versus regular functions.

Lecture 18

1 Sorting

2 Inline function

3 Reference Variable

example


```
1  #include <iostream>
2  int main()
3  {
4    using namespace std;
5    int rats = 101;
6    int & rodents = rats; // rodents is a reference-to-
         int
7      cout << "rats = " << rats;
8    cout << ", rodents = " << rodents << endl;
9    rodents++;
10   cout << "rats = " << rats;
11   cout << ", rodents = " << rodents << endl;
12
13   cout << "rats address = " << &rats;
14   cout << ", rodents address = " << &rodents << endl;
15   return 0;
16 }
```

example


```
1  #include <iostream>
2  int main()
3  {
4    using namespace std;
5    int rats = 101;
6    int & rodents = rats; // rodents is a reference
7    cout << "rats = " << rats;
8    cout << ", rodents = " << rodents << endl;
9    cout << "rats address = " << &rats;
10   cout << ", rodents address = " << &rodents << endl;
11   int bunnies = 50;
12   rodents = bunnies;// can we change the reference?
13   cout << "bunnies = " << bunnies;
14   cout << ", rats = " << rats;
15   cout << ", rodents = " << rodents <<endl;
16   cout << "bunnies address = " << &bunnies;
17   cout << ", rodents address = " << &rodents << endl;
18   return 0; }
```