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

// Function prototype
void split(const string&, char, vector<string>&);

int main()
{
   // Strings to tokenize
   string str1 = "one two three four";
   string str2 = "10:20:30:40:50";
   string str3 = "a/b/c/d/e/f";

   // vector to hold the tokens.
   vector<string> tokens;

   // Tokenize str1, using ' ' as the delimiter.
   split(str1, ' ', tokens);
   for (auto e : tokens)
      cout << e << " ";
   cout << endl;
   tokens.clear();

   // Tokenize str2, using ':' as the delimiter.
   split(str2, ':', tokens);
   for (auto e : tokens)
      cout << e << " ";
   cout << endl;
   tokens.clear();

   // Tokenize str3, using '/' as the delimiter.
   split(str3, '/', tokens);
   for (auto e : tokens)
      cout << e << " ";
   cout << endl;
   return 0;
}

//***********************************************************
// The split function splits s into tokens, using delim as the *
// delimiter. The tokens are added to the tokens vector.       *
//***********************************************************
void split(const string& s, char delim, vector<string>& tokens)
{
   int tokenStart = 0;  // Starting position of the next token

   // Find the first occurrence of the delimiter.
   int delimPosition = s.find(delim);

   // While we haven't run out of delimiters...
   while (delimPosition != string::npos)
   {
      // Extract the token.
      string tok = s.substr(tokenStart, delimPosition - tokenStart);

      // Push the token onto the tokens vector.
```

```
      tokens.push_back(tok);

      // Move delimPosition to the next character position.
      delimPosition++;

      // Move tokenStart to delmiPosition.
      tokenStart = delimPosition;

      // Find the next occurrence of the delimiter.
      delimPosition = s.find(delim, delimPosition);

      // If no more delimiters, extract the last token.
      if (delimPosition == string::npos)
      {
         // Extract the token.
         string tok = s.substr(tokenStart, delimPosition - tokenStart);

         // Push the token onto the vector.
         tokens.push_back(tok);
      }
   }
}
```