Subject: Re: Map implementation
Posted by cbpporter on Tue, 09 Apr 2019 12:57:46 GMT
View Forum Message <> Reply to Message

I re-implemented Index (copy&paste, trancode and cleanup) and should be a good starting point since it is smaller and simpler than a lot of maps out there.

I'll test mine thoroughly and benchmark, but before I benchmarked U++ vs. set vs. multiset, just to set my expectations accordingly.

U++ is pretty fast when compared to STL, but it does have exponential growth. With many items, in some tests, when multiplying items by 10, STL will go from like 250ms to 320, while Index will go from 250ms to 2 minutes :).

I used this simple test:

```
 const int SEQ = 10;
 const int MAX = 3000000 * SEQ;
 const int JUMP = 100;

 {
  Index<int> ind;

  StopWatch ts;

  for (int j = 0; j < MAX; j += JUMP)
   for (int i = j; i < j + SEQ; i++) {
    ind.Add(i);
    //ind.Debug(Cout());
   }

  Cout() << "U++ add " << ts.Elapsed() << "\n";

  ts.Reset();

  int count = 0;
  for (int i = 0; i < MAX; i++)
   if (ind.Find(i) != -1)
    count++;

  Cout() << "U++ find " << ts.Elapsed() << "\n";
  Cout() << count << "\n";
 }

 {
  std::set<int> ind;

  StopWatch ts;
```

```
 for (int j = 0; j < MAX; j += JUMP)
  for (int i = j; i < j + SEQ; i++) {
   ind.insert(i);
   //ind.Debug(Cout());
  }

 Cout() << "set add " << ts.Elapsed() << "\n";

 ts.Reset();

 int count = 0;
 for (int i = 0; i < MAX; i++)
  if (ind.find(i) != ind.end())
   count++;

 Cout() << "set find " << ts.Elapsed() << "\n";
 Cout() << count << "\n";
}

{
 std::multiset<int> ind;

 StopWatch ts;


 for (int j = 0; j < MAX; j += JUMP)
  for (int i = j; i < j + SEQ; i++) {
   ind.insert(i);
   //ind.Debug(Cout());
  }

 Cout() << "multiset add " << ts.Elapsed() << "\n";

 ts.Reset();

 int count = 0;
 for (int i = 0; i < MAX; i++)
  if (ind.find(i) != ind.end())
   count++;

 Cout() << "multiset find " << ts.Elapsed() << "\n";
 Cout() << count << "\n";
}
```

Is this an OK first artificial benchmark?

For MAX as large as the example, this is the starting point where U++ begins to loose vs. STL.

Next I need to compare memory usage...

Edit: I broke the benchmark into add and find, and the exponential growth for many items is at the find phase.
