Exemple d’utilisation de map : utiliserMap.cc

```
#include <stdio.h>
#include <map>

// exemple d'utilisation de map
// nj 2005
// g++ utiliserMap.cc
/*
 *(0 150)
 *(1 151)
 *(2 152)
 *
 */

typedef std::map<int, int> m2i;
typedef m2i::value_type m2ivt;
typedef m2i::iterator m2ii;

int main(void) {
  m2i m;
  m.insert(m2ivt(150%10, 150));
  m.insert(m2ivt(151%10, 151));
  m.insert(m2ivt(152%10, 152));
  m.insert(m2ivt(162%10, 162)); /* pas d'ajout */
  m.insert(m2ivt(171%10, 171)); /* pas d'ajout */
  m2ii i;
  for(i = m.begin(); i != m.end(); i++)
    printf("(%d %d)\n", i->first, i->second);
  printf("\n");
  return 0;
}
```

Exemple d’utilisation de map : utiliserMap2.cc

```
#include <stdio.h>
#include <map>

typedef std::map<int, int> m2i;
typedef m2i::value_type m2ivt;
typedef m2i::iterator m2ii;

// exemple d'utilisation de map
// comptage du nombre de cle dans une map
// et recherche d'un element
// nj 2005
// g++ utiliserMap2.cc
/*
 *(0 1)
 *(1 2)
 *(5 1)
 *(7 2)
 *(9 1)
 *
 *trouve : (1 2)
 */
int main(void) {
  m2i m;
  m[150%10] ++;
  m[151%10] ++;
  m[155%10] ++;
  m[157%10] ++;
  m[159%10] ++;
  m[167%10] ++;
  m[171%10] ++;
  m2ii i;
  for(i = m.begin(); i != m.end(); i++)
    printf("(%d %d)\n", i->first, i->second);
  printf("\n");
  i = m.find(1); /* recherche de l'element dont la cle est 1 */
  if(i != m.end()) /* si trouve, != de end() */
    printf("trouve : (%d %d)\n", i->first, i->second);
  return 0;
}
```

Exemple d'utilisation de map : utiliserMap3.cc

```
#include <stdio.h>
#include <map>

struct elt12 { /* element compose de 2 int */
  int c; /* c permet de comparer les elements */
  int elt;
  elt12(int _c, int _e) {c = _c; elt = _e;}
  elt12(const elt12& _e) {c = _e.c; elt = _e.elt; }
  bool operator< (const elt12&) const;
};
bool elt12::operator< (const elt12& _o) const {
  return (c < _o.c);
}
#define cleElt12(a) elt12((a), 0)

typedef std::map<elt12, int> mei;
typedef mei::value_type meivt;
typedef mei::iterator meii;

// exemple d'utilisation de map
// element de type structure
// nj 2005
// g++ utiliserMap3.cc
/*
 *(0 150 1)
 *(1 151 2)
 *(2 152 2)
 *
 *(1 151 2)
 */
```

```
int main(void) {
  mei m;
  m[elt12(150%10, 150)] ++;
  m[elt12(151%10, 151)] ++;
  m[elt12(152%10, 152)] ++;
  m[elt12(162%10, 162)] ++;
  m[elt12(171%10, 171)] ++;
  meii i;
  for(i = m.begin(); i != m.end(); i++)
    printf("(%d %d %d)\n", i->first.c, i->first.elt, i->second);
  printf("\n");
  i = m.find(cleElt12(1));
  if(i != m.end())
    printf("(%d %d %d)\n", i->first.c, i->first.elt, i->second);
  return 0;
}
```

Exemple d'utilisation de map : utiliserMap4.cc

```
#include <stdio.h>
#include <stdlib.h>
#include <map>

struct A { /* une classe avec une valeur entiere */
  int va;
  A(int _va = 0) {va = _va;};
};

typedef std::map<int, A, std::less<int> > miA;
typedef miA::value_type miAvt;
typedef miA::iterator miAii;

void affiche(int _v) { printf("affiche(%d)\n", _v); }

template <class Value>
void fonctionSpe(typename std::map<int, Value>::iterator debut,
                 typename std::map<int, Value>::iterator fin) {
  while(debut != fin) {
    affiche(debut->first); debut++;
  }
}
void fonctionNonSpe(miAii debut, miAii fin) {
  while(debut != fin) {
    affiche(debut->first); debut++;
  }
}

// exemple d'utilisation de map
// nj 2005
// g++ utiliserMap4.cc
int main(void) {
  int i, val;
  miA m;
  srand(0);
  for(i = 0; i < 10; i++) {
    val = 1 + (rand() % 100);
   m.insert(miAvt(val, A(val+1)));
  }
  printf("taille %d\n", (int)(m.size()));
  miAii j;
  for(j = m.begin(); j != m.end(); j++)
    printf("(%d A.va:%d)\n", j->first, (j->second).va);

  /* je doute de l'interet de cette fonction nommee fonctionSpe ?
   * quel est l'avantage sur fonctionNonSpe ?  */
  fonctionSpe<A>(m.begin(), m.end());
  printf("\n");
  fonctionNonSpe(m.begin(), m.end());
  return 0;
}
```

Exemple d'utilisation de map : utiliserMap5.cc

```
#include <stdio.h>
#include <string>
#include <map>

typedef std::map<std::string, int> ms;
typedef ms::value_type msvt;
typedef ms::iterator msii;

// exemple d'utilisation de map
// avec string ... (pour un hash tres lent mais generique)
// g++ utiliserMap5.cc
/*
 *(AA 0)
 *(BB 1)
 *(CC 1)
 *(DD 1)
 *
 *(AA 0)
 */
int main(void) {
  ms m;
  m.insert(msvt(std::string("AA"), 0));
  m.insert(msvt(std::string("BB"), 1));
  m.insert(msvt(std::string("CC"), 1));
  m.insert(msvt(std::string("DD"), 1));
  m.insert(msvt(std::string("AA"), 2)); // not added
  msii ii;
  for(ii = m.begin(); ii != m.end(); ii++)
    printf("(%s %d)\n", ii->first.c_str(), ii->second);
  printf("\n");
  ii = m.find("AA");
  if(ii != m.end())
    printf("(%s %d)\n", ii->first.c_str(), ii->second);
  return 0;
}
```

Exemple d'utilisation de map : utiliserMap6.cc

```
#include <stdio.h>
#include <map>

// exemple d'utilisation de map
// nj 2005
// g++ utiliserMap.cc
/*
 *(0 (10 150))
 *(1 (11 151))
 *(2 (12 152))
 *
 */
struct m_t {
  int a;
  int b;
  m_t(int _a, int _b) { a = _a; b = _b; }
};

int main(void) {
  std::map<int, m_t> m;
  {
    m_t A(10,150);
    m.insert(std::make_pair(0, A));
  }
  {
    m_t A(11,151);
    m.insert(std::make_pair(1, A));
  }
  {
    m_t A(12,152);
    m.insert(std::make_pair(2, A));
  }
  std::map<int, m_t>::iterator ii;
  for(ii = m.begin(); ii != m.end(); ii++) {
    int v1 = ii->first;
    m_t v2 = ii->second;
    printf("%d %d %d\n", v1, v2.a, v2.b);
  }
  {
    ii = m.find(1);
    m_t A(11,1510);
    ii->second = A;
  }
  printf("after update:\n");
  for(ii = m.begin(); ii != m.end(); ii++) {
    int v1 = ii->first;
    m_t v2 = ii->second;
    printf("%d %d %d\n", v1, v2.a, v2.b);
  }

  return 0;
}
```

Test avec map : testMap.cpp

```
#include <stdio.h>
#include <map>

// g++ testMap.cpp
/*
  * 1 2 0
 */
int main(void) {
  std::map<int, int> m;
  m[1] ++;
  m[2] ++;
  m[2] ++;
  fprintf(stderr, "%d %d %d\n", m[1], m[2], m[3]);
  return 0;
}
```

Test avec map : testTempsMap.cpp

```
#include <stdio.h>
#include <stdlib.h>
#include <map>
#include <ctime>
#include <sys/time.h>

struct timeval ttt_start;

void initTime() { gettimeofday(&ttt_start, 0); }
bool timeIsOver(double _t) {
  struct timeval ttt_now;
  gettimeofday(&ttt_now, 0);
  double time = ((double)(ttt_now.tv_sec - ttt_start.tv_sec)) +
    ((double)(ttt_now.tv_usec - ttt_start.tv_usec))/1000000.0;
  if(_t > time) return false;
  return true;
}

// comparaison std::map vs. tableau avec des temps fixes
/*
 * with tab --->
 *   write 6635961   1.00 sec.
 *  read  3435120   0.50 sec.
 * with map --->
 *   write 901520   1.00 sec.
 *  read  513991   0.50 sec.
 */
int main(void) {
  long int count_write = 0L;
  long int count_read = 0L;
  // must read_time < 2*write_time
  double write_time = 1.0;
  double read_time = 0.5;
  // test sur 1 sec avec TAB *****
  int t_size = 100000000; // 100M
  int* t = new int[t_size];
  srand(1);
  initTime();
  while(timeIsOver(write_time) == false) {
    int where = ((int)rand())%t_size;
    t[where] ++;
    count_write ++;
  }
  srand(1);
  initTime();
  while(timeIsOver(read_time) == false) {
    int where = ((int)rand())%t_size;
    if(t[where] >= 1) count_read ++;
  }
  printf("with tab --->\n");
  printf("  write %ld   %.2f sec.\n", count_write, write_time);
  printf("  read  %ld   %.2f sec.\n", count_read, read_time);
  fflush(stdout);
  delete[] t;
  count_write = 0L;
  count_read = 0L;

  // test sur 1 sec avec MAP *****
  srand(1);
  initTime();
  std::map<int, int> m;
  while(timeIsOver(write_time) == false) {
    int where = (int)rand();
    m[where] ++;
    count_write ++;
  }

  srand(1);
  initTime();
  while(timeIsOver(read_time) == false) {
    int where = (int)rand();
    if(m[where] >= 1) count_read ++;
  }
  printf("with map --->\n");
  printf("  write %ld   %.2f sec.\n", count_write, write_time);
  printf("  read  %ld   %.2f sec.\n", count_read, read_time);
  fflush(stdout);

  return 0;
}
```