```
=== 演習1 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex06-1.cpp
$ ./a.out
{1, 5} {2, 6} {3, 7} {4, 8}
{3, 7}


--- ex06-1.cpp ---
// リストだけができること
#include <iostream>
#include <list>

struct Point {
   int x, y;
};

int main()
{
   std::list<Point> pts{ {3,7} };
   auto org { pts.begin() };

   pts.push_front( {1,5} );
   pts.insert(org, {2, 6} );
   pts.push_back( {4, 8} );

   for (auto& p : pts)
      std::cout <<"{"<< p.x <<", "<< p.y <<"} ";
   std::cout <<"\n{"<< org->x <<", "<< org->y <<"}\n";
}
```

```
=== 演習2 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex06-2.cpp
$ ./a.out
300


--- ex06-2.cpp ---
// 連想配列としてのstd::map
#include <iostream>
#include <map>
using std::cout, std::string;

int main()
{
   std::map<string,int> price;

   price["orange"] = 150;
   price["banana"] = 300;
   price["pineapple"] = 850;

   cout << price["banana"] <<"\n";
}
```

```
=== 演習3 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex06-3.cpp
$ ./a.out
0
4


--- ex06-3.cpp ---
// 連想配列としてのstd::map
#include <iostream>
#include <map>
using std::cout, std::string;

int main()
{
   std::map<string,int> price;

   price["orange"] = 150;
   price["banana"] = 300;
   price["pineapple"] = 850;

   cout << price["nabana"] <<"\n";
   cout << price.size() <<"\n";
}
```

```
=== 演習4 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex06-4.cpp
$ ./a.out
3


--- ex06-4.cpp ---
// 連想配列としてのstd::map
// 削除
#include <iostream>
#include <map>

int main()
{
   std::map<std::string,int> price {
      {"orange",150},
      {"tomato",120},
      {"banana",300},
      {"pineapple",450} };

   // 削除にはイテレータが必要
   if (auto it{price.find("banana")}; it != price.end())
      price.erase(it);

   std::cout << price.size() <<"\n"; // 3
}
```

```
=== 演習5 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex06-5.cpp
$ echo a b c a a b c d | ./a.out
a b c d
$ echo abc bbc ccba abc abc d | ./a.out
abc bbc ccba d


--- ex06-5.cpp ---
// set
#include <iostream>
#include <set>
using std::string;

int main()
{
   std::set<string> a;

   for (string s; std::cin >> s; )
      a.insert(s);

   for (auto e : a)
      std::cout << e <<" ";
   std::cout<<"\n";
}
```

```
=== 演習6 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex06-6.cpp
$ ./a.out
banana : 300
tomato : 120
pineapple : 450
orange : 150


--- ex06-6.cpp ---
// unordered_map
#include <iostream>
#include <unordered_map>
using std::string;
int main()
{
   std::unordered_map<string,int> price {
      {"orange", 150},
      {"tomato", 120},
      {"banana", 300},
      {"pineapple", 450}
   };

   for (auto& [n,p] : price)
      std::cout << n <<" : "<< p <<"\n";
}
```

```
=== 演習7 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex06-7.cpp
$ ./a.out
6 10 4 5 9 8 2 1 3 7
$ ./a.out
3 1 6 2 4 10 7 5 9 8
$ ./a.out 100
60 23 30 62 99 43 64 22 98 61 27 76 56 28 69 55 12 49 65 39 53 2 89 97 58 45 19 95 71 40
 92 57 74 84 38 90 87 15 81 7 18 51 6 52 72 9 48 80 25 46 5 91 41 4 67 3 88 100 73 1 24
77 59 10 66 93 21 44 35 29 68 8 16 42 20 54 11 17 85 70 50 94 14 78 75 33 26 82 83 13 34
 96 32 47 37 79 31 63 86 36
$ ./a.out 100
86 63 77 94 52 69 97 41 57 8 34 36 96 31 14 68 70 35 92 49 5 29 81 38 50 66 18 12 40 30
26 64 37 22 82 42 4 76 55 16 15 79 84 10 59 7 90 58 47 91 72 24 33 62 27 56 75 83 100 73
 17 74 98 43 20 11 95 44 61 6 88 67 93 60 39 13 3 9 2 19 32 28 54 80 45 78 51 46 71 1 21
 65 48 53 89 85 23 87 25 99


--- random.hpp ---
// 乱数生成
// https://ja.wikipedia.org/wiki/メルセンヌ・ツイスタ
#include <random>
class UniDist {
   std::random_device seed;
   std::mt19937 engine;
   //std::default_random_engine engine{}; // for debug
   std::uniform_int_distribution<int> udist;
public:
   UniDist(int first, int last)
      :seed{}, engine{seed()}, udist{first,last}{}
   // [first, last] の一様乱数
   auto get(){ return udist(engine); }
};

class ExpDist{ // unit時間にtimes回発生する条件の乱数
   std::random_device seed;
   std::mt19937 engine;
   std::exponential_distribution<double> edist;
   double unit;
public:
   ExpDist(double lambda, double u =1.0)
      :seed{}, engine{seed()}, edist{lambda}, unit{u}{}
   // 次に起こるまでの時間
   auto get(){ return edist(engine)*unit; }
};


--- ex06-7.cpp ---
// 重複のない乱数の列
#include <iostream>
#include <set>
#include "random.hpp"

void rndseq_set(int n)
{
   std::set<int> chk;
   UniDist r{1,n};
   for (int i=0; i<n; i++) {
      int x{};
      while (true) {
         x = r.get();
         if (auto [it,f] {chk.insert(x)}; f)
            break;
      }
      std::cout << x <<" ";
   }
}

int main(int argc, char *argv[])
```

```
{
   int n { argc > 1 ? std::atoi(argv[1]):10 };
   rndseq_set(n);
   std::cout <<"\n";
}
```

```
=== 演習8 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex06-8.cpp
$ cat poem.txt
I Know Why The Caged Bird Sings
Maya Angelou

The free bird leaps
on the back of the wind
and floats downstream
till the current ends
and dips his wings
in the orange sun rays
and dares to claim the sky.

But a bird that stalks
down his narrow cage
can seldom see through
his bars of rage
his wings are clipped and
his feet are tied
so he opens his throat to sing.

The caged bird sings
with fearful trill
of the things unknown
but longed for still
and his tune is heard
on the distant hill for the caged bird
sings of freedom

The free bird thinks of another breeze
and the trade winds soft through the sighing trees
and the fat worms waiting on a dawn-bright lawn
and he names the sky his own.

But a caged bird stands on the grave of dreams
his shadow shouts on a nightmare scream
his wings are clipped and his feet are tied
so he opens his throat to sing

The caged bird sings
with a fearful trill
of things unknown
but longed for still
and his tune is heard
on the distant hill
for the caged bird
sings of freedom.

$ ./a.out < poem.txt
Angelou: 1
Bird: 1
But: 2
Caged: 1
I: 1
Know: 1
Maya: 1
Sings: 1
The: 5
Why: 1
a: 5
and: 10
another: 1
are: 4
back: 1
bars: 1
bird: 8
breeze: 1
but: 2
```

cage: 1
caged: 5
can: 1
claim: 1
clipped: 2
current: 1
dares: 1
dawn-bright: 1
dips: 1
distant: 2
down: 1
downstream: 1
dreams: 1
ends: 1
fat: 1
fearful: 2
feet: 2
floats: 1
for: 4
free: 2
freedom: 2
grave: 1
he: 3
heard: 2
hill: 2
his: 13
in: 1
is: 2
lawn: 1
leaps: 1
longed: 2
names: 1
narrow: 1
nightmare: 1
of: 8
on: 6
opens: 2
orange: 1
own: 1
rage: 1
rays: 1
scream: 1
see: 1
seldom: 1
shadow: 1
shouts: 1
sighing: 1
sing: 2
sings: 4
sky: 2
so: 2
soft: 1
stalks: 1
stands: 1
still: 2
sun: 1
that: 1
the: 15
things: 2
thinks: 1
throat: 2
through: 2
tied: 2
till: 1
to: 3
trade: 1
trees: 1
trill: 2
tune: 2
unknown: 2
waiting: 1

```
wind: 1
winds: 1
wings: 3
with: 2
worms: 1


--- poem.txt ---
I Know Why The Caged Bird Sings
Maya Angelou

The free bird leaps
on the back of the wind
and floats downstream
till the current ends
and dips his wings
in the orange sun rays
and dares to claim the sky.

But a bird that stalks
down his narrow cage
can seldom see through
his bars of rage
his wings are clipped and
his feet are tied
so he opens his throat to sing.

The caged bird sings
with fearful trill
of the things unknown
but longed for still
and his tune is heard
on the distant hill for the caged bird
sings of freedom

The free bird thinks of another breeze
and the trade winds soft through the sighing trees
and the fat worms waiting on a dawn-bright lawn
and he names the sky his own.

But a caged bird stands on the grave of dreams
his shadow shouts on a nightmare scream
his wings are clipped and his feet are tied
so he opens his throat to sing

The caged bird sings
with a fearful trill
of things unknown
but longed for still
and his tune is heard
on the distant hill
for the caged bird
sings of freedom.



--- ex06-8.cpp ---
// 単語の出現頻度
#include <iostream>
#include <cctype>
#include <map>

auto normalize(std::string s)
{
   if (!std::isalpha(s.back()))
      s = s.substr(0,s.size()-1);
   return s;
}

int main()
{
```

```
   std::map<std::string,int> words;
   for (std::string s; std::cin >> s; )
      ++ words[ normalize(s) ];
   for (auto [w,c]: words)
      std::cout << w <<": "<< c <<"\n";
}
```