```
=== 演習1 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex12-1.cpp
$ ./a.out
a:0 c:1 a:3 a:4 c:8 a:10 c:11 a:13 c:14
b:2 b:5 b:7 b:12
c:1 b:2 b:5 b:7 c:8 c:11 b:12 c:14


--- ex12-1.cpp ---
// すべての要素の添字
#include <iostream>

using std::cout, std::string;

void find_all(string src, string sub)
{
   size_t idx {src.find_first_of(sub, 0)};
   while ( idx != string::npos ) {
      cout << src[idx] <<":"<< idx <<" ";
      ++idx; // 次の要素
      idx = src.find_first_of(sub, idx);
   }
   cout <<"\n";
}

int main()
{
   string a {"acbaabebcdacbac"};
   find_all(a, "ac");
   find_all(a, "b");
   find_all(a, "cb");
}
```

```
=== 演習2 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex12-2.cpp
$ ./a.out 'a[bc]c'
"abc" "acc"


--- ex12-2.cpp ---
#include <iostream>
#include <vector>
#include <regex>
using std::cout, std::string;
int main(int argc, char *argv[])
{
   if (argc < 2) {
      cout <<"usage: "<< argv[0] <<" regex\n";
      return 1;
   }
   string rs { argv[1] };
   std::regex r { rs };
   std::vector<string> vs { "ac", "abc", "acc",
   "a c", "a0c", "a1c", "abd", "a3d", "axc", "a\nc"};
   for (auto e:vs)
      if (std::regex_match(e, r))
         cout <<"\""<< e <<"\" ";
   cout <<"\n";
}
```

```
=== 演習3 解答例 ============================
--- run.txt ---
$ ./a.out 'a\dc'
"a0c" "a1c"
以下のでも良い
$ ./a.out 'a[0-9]c'
"a0c" "a1c"

$ ./a.out 'a.c'
"abc" "acc" "a c" "a0c" "a1c" "axc"
$ ./a.out '.*c$'
"ac" "abc" "acc" "a c" "a0c" "a1c" "axc"
$ ./a.out 'ab?c'
"ac" "abc"
$ ./a.out 'a\Sc'
"abc" "acc" "a0c" "a1c" "axc"
```

```
=== 演習4 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex12-4.cpp
$ ./a.out 'a.*c'
ac, aaac, ac ccc,
ac, aaac,  aac , ac ccc, anc ,
[okam@csz840 ex4]$ ./a.out 'a[a-z]c'

aaac,  aac , anc ,
[okam@csz840 ex4]$ ./a.out 'a[a-z]*c'
ac, aaac,
ac, aaac,  aac , ac ccc, anc ,


--- ex12-4.cpp ---
#include <iostream>
#include <vector>
#include <regex>
using std::cout, std::string;
int main(int argc, char* argv[])
{
   if (argc < 2) {
      cout <<"usage: "<< argv[0] <<" regex\n";
      return 1;
   }
   string rs {argv[1]};
   std::regex r{rs}; // 正規表現
   std::vector<string> lst // 検索対象文字列群
   {"ac", "aaac", " aac ", "akb", "ac ccc", "anc "};

   for (auto& s : lst)
      if (std::regex_match(s, r)) // 全体一致
         cout << s <<", ";
   cout <<"\n";

   for (auto& s : lst)
      if (std::regex_search(s, r)) // 部分一致
         cout << s <<", ";
   cout <<"\n";
}
```

```
=== 演習5 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex12-5.cpp
$ ./a.out 'ak([ abc]*)c'
data: acaaac aac akbac cccancnkp
m.prefix():acaaac aac
m.suffix():ancnkp
m[0]:akbac ccc, m.position(0):11
m[1]:bac cc, m.position(1):13
$ ./a.out 'a{3}(\w)'
data: acaaac aac akbac cccancnkp
m.prefix():ac
m.suffix(): aac akbac cccancnkp
m[0]:aaac, m.position(0):2
m[1]:c, m.position(1):5



--- ex12-5.cpp ---
// smatch detail
// ak([ abc]*)c
#include <iostream>
#include <regex>
using std::cout, std::string;
int main(int argc, char* argv[])
{
   if (argc < 2) {
      cout <<"usage: "<< argv[0] <<" regex\n";
      return 1;
   }
   string rs {argv[1]};
   string data {"acaaac aac akbac cccancnkp"};
   cout <<"data: " << data <<"\n";
   std::regex r{ rs };
   std::smatch m;
   std::regex_search(data, m, r);
   if (!m.empty()) {
      cout <<"m.prefix():"<< m.prefix()<<"\n"
           <<"m.suffix():"<< m.suffix()<<"\n";
      for (size_t i = 0; i < m.size(); i++) {
         cout <<"m["<< i <<"]:"<< m[i] <<", "
              <<"m.position("<< i <<"):"
              << m.position(i) <<"\n";
      }
   }  // stringを得るには\verb|.str()|
}
```

```
=== 演習6 解答例 ============================
--- compile-and-run.txt ---
$ g++ -std=c++17 ex12-6.cpp
$ ./a.out '\b(aac).*k'
data: acaaac aac akbac cccancnkp
m.prefix():acaaac
m.suffix():p
m[0]:aac akbac cccancnk, m.position(0):7
m[1]:aac, m.position(1):7
$ ./a.out 'a.*(a.c).*(\1).*c'
data: acaaac aac akbac cccancnkp
m.prefix():
m.suffix():nkp
m[0]:acaaac aac akbac cccanc, m.position(0):0
m[1]:aac, m.position(1):3
m[2]:aac, m.position(2):7


--- ex12-6.cpp ---
// smatch detail
// ak([ abc]*)c
#include <iostream>
#include <regex>
using std::cout, std::string;
int main(int argc, char* argv[])
{
   if (argc < 2) {
      cout <<"usage: "<< argv[0] <<" regex\n";
      return 1;
   }
   string rs {argv[1]};
   string data {"acaaac aac akbac cccancnkp"};
   cout <<"data: " << data <<"\n";
   std::regex r{ rs };
   std::sregex_iterator pos{data.cbegin(), data.cend(), r};
   std::sregex_iterator end;
   for ( ; pos != end; ++pos){
      auto m{*pos}; // std::smatch
      cout <<"m.prefix():"<< m.prefix()<<"\n"
           <<"m.suffix():"<< m.suffix()<<"\n";
      for (size_t i = 0; i < m.size(); i++) {
         cout <<"m["<< i <<"]:"<< m[i] <<", "
              <<"m.position("<< i <<"):"
              << m.position(i) <<"\n";
      }
   }  // stringを得るには\verb|.str()|
}
```