Appendix

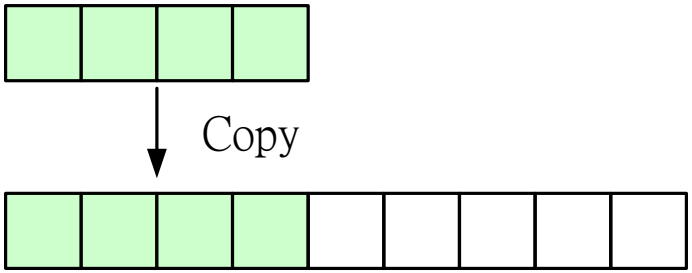
C / C++ / Java Library

Containers in Java

List	ArrayList LinkedList	a variable-length array a linked list
Set	TreeSet HashSet	a sorted set implemented by tree a un-sorted set implemented by hash table
Map	TreeMap HashMap	a dictionary (sorted keys) a dictionary (un-sorted keys, faster)
Iterator		Pointer to container

- ArrayList

add(Object) addAll(Collection)	add an object add a set of objects
clear()	
contains(Object) containsAll(Collection)	check if an object exists check if a set of objects exist
iterator()	
remove(Object) removeAll(Collection)	remove an object remove a set of objects
get(int)	works like an array
set(int, Object)	works like an array
toString()	



- ArrayList

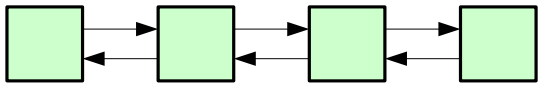
- Example:

```
public static void main(String[] args) {
  Scanner cin = new Scanner(System.in);
  ArrayList<Integer> list = new ArrayList<Integer>();
  int x;
  while ((x = cin.nextInt()) != 0) {
    list.add(x);
  }

  System.out.println("total " + list.size() + " input elements.");
  System.out.println(list.toString());
  System.out.println("2nd element = " + list.get(2));
}
```

```
3 5 7
1 0

total 4 input elements.
[3, 5, 7, 1]
2nd element = 7
```

- LinkedList

addFirst(Object)	add an object to be the first element
addLast(Object) add(Object)	add an object to be the last element
removeFirst(Object) removeLast(Object)	remove the first element remove the last element
getFirst() getLast()	get the first element get the last element
get(int) set(int, Object)	like the ArrayList (but very slow!!)



- Set (TreeSet, HashSet)

TreeSet can output a sorted iterator.
O(log n)-time operators

add(Object) addAll(Collection)	add an object add a set of objects
clear()	
isEmpty()	
contains(Object) containsAll(Collection)	check if an object exists check if a set of objects exist
iterator()	
remove(Object) removeAll(Collection)	remove an object remove a set of objects
toString()	

TreeSet

- Example:

```
public static void main(String[] args) {
  Scanner cin = new Scanner(System.in);
  Set<Integer> s = new TreeSet<Integer>();
  while (cin.hasNextInt()) {
    s.add(cin.nextInt());
  }

  for (Iterator iter = s.iterator(); iter.hasNext();) {
    System.out.print(iter.next() + " ");
  }
  System.out.println();

  System.out.println(s.toString());
  System.out.println(s.contains(0));
}
```

```
・Input
3 5 7 1

・Output
1 3 5 7
[1, 3, 5, 7]
false
```

- Map (TreeMap, HashMap)

TreeMap can output a sorted iterator on key set.
O(log n)-time operators

add(Object) addAll(Collection)	add an object add a set of objects
clear()	
isEmpty()	
contains(Object) containsAll(Collection)	check if an object exists check if a set of objects exist
iterator()	
remove(Object) removeAll(Collection)	remove an object remove a set of objects
toString()	

- TreeMap

- Example:

```
public static void main(String[] args) {
  Map<String, Integer> map = new TreeMap<String, Integer>();

  for (int i = 0; i < args.length; i++) {
    map.put(args[i], i);
  }

  for (Iterator<String> iter = map.keySet().iterator(); iter.hasNext();) {
    String key = iter.next();
    System.out.println(key + " -> " + map.get(key));
  }
}
```

```
C:\myjava>java TestMap John Tom Mary Andy
Andy -> 3
John -> 0
Mary -> 2
Tom -> 1
```

Containers in C++

- Compare of Java and C++:

	Java	C++
List	ArrayList LinkedList	vector list deque queue stack
Set	TreeSet HashSet	set multi-set
Map	TreeMap HashMap	map multi-map

- #include <vector>

- Example:

```
vector<int> v;
int x;
while (cin >> x)
  v.push_back(x);

cout << "total " << v.size() << " input elements. “ << endl;
reverse(v.begin(), v.end());
for (int i = 0; i < v.size(); i++)
  cout << v[i] <<  “, ";
```

```
3 5 7
1 0

total 5 input elements.
0, 1, 7, 5, 3,
```

- #include <set>

- Example:

```
set<string> s;
s.insert("xyz");
s.insert("def");
s.insert("abc");
s.insert("bbb");
s.insert("bbb");

set<string>::iterator iter;
for( iter = s.begin(); iter != s.end(); iter++ ) {
  cout << *iter << endl;
}
cout << s.count("aaa") << endl;
cout << s.count("bbb") << endl;
```

```
abc
bbb
def
xyz
0
1
```

- #include <map>

- Example:

```
map<string,int> stringCounts;

string str;
while (cin >> str) stringCounts[str]++;

map<string,int>::iterator iter;
for( iter = stringCounts.begin(); iter != stringCounts.end(); iter++ ) {
  cout << iter->first << "=" << iter->second << endl;
}
```

```
here are some words and here are some more words

and=1
are=2
here=2
more=1
some=2
words=2
```

Sorting / Searching in C

qsort	Quick sort	stdlib.h
bsearch	Binary search	stdlib.h

void qsort(void *base, size_t num, size_t width, __cdecl *compare);
void *bsearch(void *key, void *base, size_t nelem, size_t size, __cdecl *compare);

- Prepare a **compare** function
  int compare (const void * elem1, const void * elem2);

return value	description
<0	*elem1 < *elem2
0	*elem1 == *elem2
>0	*elem1 > *elem2

- Example:

```
int compare(const void *arg1, const void *arg2 ) {
  char **a = (char **)arg1;
  char **b = (char **)arg2;
  return _strcmpi(*a, *b);
}

int main( int argc, char **argv ) {
   char **result;
   char *key = "DOG";
   int i;

   qsort(argv, argc, sizeof(char *), compare);
   for( i = 0; i < argc; ++i )    /* Output sorted list */
      printf( "%s ", argv[i] );

   result = bsearch(&key, argv, argc, sizeof(char *), compare);
   if( result )
      printf( "\n%s found at %Fp (%d)\n", *result, result, (result - argv));
   else
      printf( "\nDOG not found!\n" );
}
```

```
C:\work>mytest DOG PIG HORSE CAT HUMAN RAT COW GOAT
CAT COW DOG GOAT HORSE HUMAN PIG RAT mytest
DOG found at 003D2530 (2)
```

Sorting / Searching in C++

sort	Quick sort	<algorithm>
binary_search	Binary search	<algorithm>

- Can be applied on both arrays and STL vectors

```
template<class RanIt>
    void sort(RanIt first, RanIt last);
template<class RanIt, class Pr>
    void sort(RanIt first, RanIt last, Pr pred);

template<class FwdIt, class Ty>
    bool binary_search(FwdIt first, FwdIt last, const Ty& val);
template<class FwdIt, class Ty, class Pr>
    bool binary_search(FwdIt first, FwdIt last, const Ty& val, Pr pred);
```

- Example:

```
#include <algorithm>
#include <vector>
#include <iostream>
#include <iterator>

using namespace std;

int main() {
  int ia[] = {29,23,20,22,17,15,26,51,19,12,35,40};
  vector<int> vec(ia, ia+12);

  sort(&ia[0], &ia[12]);
  bool found_it = binary_search(&ia[0], &ia[12], 18);
  copy(ia, ia+12, ostream_iterator<int, char>(cout, " ")), cout << endl;

  sort(vec.begin(), vec.end(), greater<int>());
  found_it = binary_search(vec.begin(), vec.end(), 26, greater<int>());
  copy(vec.begin(), vec.end(), ostream_iterator<int, char>(cout, " ")), cout << endl;

  if (found_it)  // is only a bool
    cout << "binary_search(): success!\n";
}
```

```
12 15 17 19 20 22 23 26 29 35 40 51
51 40 35 29 26 23 22 20 19 17 15 12
binary_search(): success!
```

Sorting / Searching in Java

Arrays.sort() Collections.sort()	Sorting on array / List
Arrays.binarySearch() Collections.binarySearch()	Searching on array / List (if not found, return the index to insert)
Collections.reverse()	Reverse the List
Arrays.asList()	Treat an array as a List object

- Example:

```
int[] arr1 = {5, 7, 1, 3, 9};
Arrays.sort(arr1);
System.out.println(Arrays.toString(arr1));
System.out.printf("index of %d is %d\n", 3, Arrays.binarySearch(arr1, 3));
System.out.printf("index of %d is %d\n", 4, Arrays.binarySearch(arr1, 4));
```

```
[1, 3, 5, 7, 9]
index of 3 is 1
index of 4 is -3      // -(-3 + 1) = 2
```

- Example:

```
String[] arr1 = {"ABCD", "BADC", "DCBA", "ADCB"};
System.out.println(Arrays.toString(arr1));

Arrays.sort(arr1);
System.out.println(Arrays.toString(arr1));

System.out.printf("index of %s is %d\n", "BADC", Arrays.binarySearch(arr1, "BADC"));

Collections.reverse(Arrays.asList(arr1));
System.out.println(Arrays.toString(arr1));

System.out.printf("index of %s is %d\n", "BADC", Arrays.binarySearch(arr1, "BADC"));
```

```
[ABCD, BADC, DCBA, ADCB]
[ABCD, ADCB, BADC, DCBA]
index of BADC is 2
[DCBA, BADC, ADCB, ABCD]
index of BADC is 1
```

- Example (10905):

```
public static void main(String[] args) {
  String[] data = {"123", "56", "90", "124", "9"}
  Arrays.sort(data, new Cmp());
  System.out.println(Arrays.toString(arr1));
}

public static class Cmp implements Comparator<String> {
  public int compare(String s1, String s2) {
    String new_s1 = s1 + s2;
    String new_s2 = s2 + s1;
    return - new_s1.compareTo(new_s2);
  }
}
```

```
9 90 56 124 123
```

Very Big Integer (Java)

new BigInteger(String) BigInteger.valueOf(long)	String / int ➔ BigInteger
.toString() .intValue()	BigInteger ➔ String / int
.add() .subtract() .multiply() .divide() .mod() .remainder()	arithmetic operators
.gcd() .pow() .isProbablePrime()	special arithmetic operators
.and() .or() .not() .xor() .setBit() .shiftLeft()	bit-wise operators

- Example:

```
static BigInteger factorial(int n) {
  BigInteger result = BigInteger.valueOf(1);
  for (int i = 2; i <= n; i++) {
    result = result.multiply(BigInteger.valueOf(i));
  }
  return result;
}

public static void main(String[] args) {
  System.out.printf("%d! = %s\n", 100, factorial(100).toString());
}
```

```
100! = 93326215443944152681699238856266700490715968264381621468592963895217
599993229915608941463976156518286253697920827223758251185210916864000000000
000000000000000
```

Permutation in C++

next_permutation	Next permutation state	<algorithm>
prev_permutation	Previous permutation state	<algorithm>

```
#include <algorithm>

bool next_permutation(BidirectionalIterator first, BidirectionalIterator last, [Compare]);
bool prev_permutation(BidirectionalIterator first, BidirectionalIterator last, [Compare]);
```

```
#include <algorithm>
#include <iostream>
using namespace std;

bool nameCompare(char * a, char * b) { return strcmp(a, b) <= 0; }

void main () {
  int start [] = { 1, 2, 3};
  do
     copy (start, start + 3, ostream_iterator<int,char> (cout, " ")), cout << endl;
  while (next_permutation(start, start + 3));

  char * words [] = {"Alpha", "Beta", "Gamma"};
  do
     copy (words, words + 3, ostream_iterator<char *,char> (cout, " ")), cout << endl;
  while (next_permutation(words, words + 3, nameCompare));
}
```

```
1 2 3
1 3 2
2 1 3
2 3 1
3 1 2
3 2 1
Alpha Beta Gamma
Alpha Gamma Beta
Beta Alpha Gamma
Beta Gamma Alpha
Gamma Alpha Beta
Gamma Beta Alpha
```

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

void main () {
  vector<char> word(4);
  word[0] = 'b'; word[1] = 'e'; word[2] = 'l'; word[3] = 'a';

  do
     copy( word.begin(), word.end(), out_stream ), cout << endl;
  while (prev_permutation(word.begin(), word.end()));
  cout << endl;
}
```

```
b e l a
b e a l
b a l e
b a e l
a l e b
a l b e
a e l b
a e b l
a b l e
a b e l
```

Debug (C / C++ / Java)

- Assertion:
  Things that all team members should know and obey.
  Bugs made by inattention or stupid team members.

C	#include <assert.h>
C++	#include <cassert>
Java	build-in in J2SE 5.0 (JDK 1.5)

- Example:

```
int factorial(int n) {
  int result = 1;
  for (int i = 2; i <= n; i++) {
    result = result * i;
  }

  return result;
}
```

(-3)! = ??

13! will overflow!!!

- -1 means error occurs

```
int factorial(int n) {
   // 13! will overflow
  if (n >= 0 && n <= 12)
    return -1;

  int result = 1;
  for (int i = 2; i <= n; i++) {
    result = result * i;
  }

  return result;
}
```

- Using assertion

```
#include <assert.h>

int factorial(int n) {
  // 13! will overflow
  assert(n >= 0 && n <= 12);

  int result = 1;
  for (int i = 2; i <= n; i++) {
    result = result * i;
  }

  return result;
}
```

```
Assertion failed: n >= 0 && n <= 12, file C:\test.cpp, line 11

This application has requested the Runtime to terminate it in an unusual way.
Please contact the application's support team for more information.
```

String (char[]) ⇔ Number (C / C++)

atoi, atof, atol	char * ➔ int / double / long	stdlib.h
itoa	int ➔ char *	stdlib.h
sscanf	alternate of atoi (in Unix)	stdio.h
sprintf	alternate of itoa (in Unix)	stdio.h

- Example (C):

```
char *s1 = "12345";
char s2[6];

int n = atoi(s1);
sscanf(s1, "%d", &n);

itoa(n, s2, 10);
sprintf(s2, "%d", n);
```

- Example (C++):

```
string s1 = "12345";
char s2[6];

int n = atoi(s1.c_str());

itoa(n, s2, 10);
s1 = s2;  // string <- char *
```

String ⇔ Number (Java)

Integer.parseInt() Double.parseDouble() Long.parseLong()	String ➔ int / double / long
String.valueOf()	int / double / long ➔ String

- Example:

```
String s1 = "123.45";
String s2;

double n = Double.parseDouble(s1);
s2 = String.valueOf(n);
```

I/O in C++

- 下列的輸入格式

1. Test case 的數量未定
2. Test case 第一個為 N
3. 後面接著 N 個 integer (中間可任意換行)
4. 讀到 N=0 時結束

$$N\ a_1\ a_2\ a_3 \ldots a_N$$

- Example (good input):

```
5
123 215 864 365 754
4
657 624 987 456
0
```

- Example (bad input):

```
5 123
215
864 365 754
4 657
624 987 456
0
```

- Example:

```
int N;
int data[MAX]; // view as integer

while (cin >> N) {
  for (int i = 0; i < N; i++) {
    cin >> data[i];
  }
}
```

- Example:

```
int N;
string data[MAX];    // view as string

while (cin >> N) {
  for (int i = 0; i < N; i++) {
    cin >> data[i];
  }
}
```

cin.getline()	Get a line from the stdin stream.
setw(int n)	Set the width of the next output to n chars.
setprecision(int n)	Set the number of decimal fractions of the next input to n chars.

- Example:

```
char name1[256];
string name2;

cout << "Enter your name: ";
cin.getline(name1,256);
getline(cin, name2, "\n");

double d;
cin >> d;
cout << fixd << setprecision(3) << setw(10) << d;
printf("%10.3f", d);
```

String Manipulation in C

- strcat : Append a string.

```
#include <string.h>
char *strcat( char *strDestination, const char *strSource );
```

- Example:

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

void main() {
   char s[100] = "Hello!";
   strcat(s, " World!");
   printf("%s", s);
}
```

```
Hello! World!
```

- strchr : Find a character in a string.

```
#include <string.h>
char *strchr( const char *string, int c );
```

- Example:

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

void main() {
   char *s = "This is a sample!";

   char *p1 = strchr(s, 's');
   char *p2 = strchr(p1 + 1, 's');
   printf("%s\n%s\n%s", s, p1, p2);
}
```

```
This is a sample!
s is a sample!
s a sample!
```

- strcmp : Compare strings.

#include <string.h>
int strcmp(const char *string1, const char *string2);

- Example:

```
void main() {
   char *s1 = "This is a sample";
   char *s2 = "This is another sample";

   int result = strcmp(s1, s2);
   if (result < 0)
      printf("s1 < s2");
   if (result == 0)
      printf("s1 == s2");
   if (result > 0)
      printf("s1 > s2");
}
```

```
s1 < s2
```

- strcpy : Copy a string.

#include <string.h>
char *strcpy(char *strDestination, const char *strSource);

- Example:

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

void main() {
   char s1[80] = "This is a sample";
   char s2[80];

   strcpy(s2, s1);
   printf("s1 = %s\ns2 = %s", s1, s2);
}
```

```
s1 = This is a sample
s2 = This is a sample
```

- strlen : Get the length of a string.

```
#include <string.h>
size_t strlen( const char *string );
```

- Example:

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

void main() {
   char s1[80] = "This is a sample";

   int len = strlen(s1);
   printf("s1 = %s\nlength of s1 = %d", s1, len);
}
```

```
s1 = This is a sample
length of s1 = 16
```

- strstr : Find a substring.

#include <string.h>
char *strstr(const char *string, const char *strCharSet);

- Example:

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

void main() {
   char *s = "This is a sample";
   char *sub = "is a";
   char *alt = "The";

   char *sp = strstr(s, sub);
   char *ap = strstr(s, alt);

   printf("find %s : %s\n", sub, sp);
   printf("find %s : %s\n", alt, ap);
}
```

```
find is a : is a sample
find The : (null)
```

- strtok : Find the next token in a string.

#include <string.h>
char *strtok(char *strToken, const char *strDelimit);

- Example:

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

void main() {
   char *s = "This\tis a sample";
   char *token;

   token = strtok(s, " \t");
   while (token != NULL)
   {
      printf("%s\n", token);
      token = strtok(NULL, " \t");
   }
}
```

```
This
is
a
sample
```

String Manipulation in C++

- Compare of C and C++:

C	C++
char * strcat(char *, const char *);	string1 + string2
char * strchr(const char *, int);	string1.find(ch1)
char * strcpy(char *, const char *);	string1 = string2
int strcmp(const char *, const char *);	string1.compare(string2) string1.compare(index, len, string2)
size_t strlen(const char *);	string.length()

char * _strlwr(char *);	for (i = 0; i<string1.length(); i++) {
	string1[i] = tolower(string1[i]);
	}

char * _strupr(char *)	for (i = 0; i<string1.length(); i++) {
	string1[i] = toupper(string1[i]);
	}

char * strstr(const char *, const char *);	string1.find(string2) string1.rfind(string2)
char * strtok(char *, const char *);	string1.find_first_of(string2)

- insert : insert a substring into a string.

#include <string>
using namespace std;
basic_string& insert(size_type pos1, const basic_string& s);
basic_string& insert(size_type pos, const basic_string& s, size_type pos2 = 0, size_type n = npos);

- Example:

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

void main() {
  string text, word;
  text = "This is a sample";
  word = "very good ";
  text.insert(10,word,5,5);
  cout<<text<<endl;
}
```

```
This is a good sample
```

- erase : Remove some chars from a string.

#include <string>
using namespace std;
basic_string& erase (size_type pos = 0, size_type n = npos);

- Example:

```
#include <string>
#include <stdio.h>
using namespace std;

void main() {
  string text;
  text="abcdefghi";
  text.erase(4,3);
  cout<<text<<endl;
}
```

```
abcdhi
```