A. KNOWLEDGE AND TOOLS

1. string	C++ string	C string
(1) Operations		
#include	<string>	<cstring>
declaration	string s; // automatically an empty string	char s[LINES_NUM][LENGTH+1]; // if no further initialization, this c string is undefined
initialization	string s="OK";	char s[]="OK"; char s[]={'O','K','\0'};

assignment/ copy	s="ok";	~~s="ok";~~ //wrong.
		//if already initialized, need to replace every char by using loop (and don't forget the '\0' end)
		//if not initialized but already declared, use **strcpy(s,** "ok"**);**

concatenation/append 1	s1+=s2;	**strcat(s1,s2);** // no '\0' at the end of s1 anymore
concatenation/append 2	s=s1+s2;	strcpy(str, strcat(str1, str2));

comparison	s1<s2; // return a bool
	s1=s2; // return a bool
	s1>s2; // return a bool
		if (**strcmp(s1, s2)<0**);
		if (**strcmp(s1, s2)==0**);
		if (**strcmp(s1, s2)>0**);

length/size	**s.size();** s.length();	**strlen(s);**

if empty	**s.empty();** // return a bool
	s.size()==0;
	s.length()==0;
	s==""";
		s[0]=='\0';

visit every character in a string s and copy all the nonblank characters to a new string t	for (size_t k=0; k!=s.size(); k++){
	  if (s[k]!=' ') { // if not a blank
	  t+=s[k];
	  }
	}
		int j=0;
		for (size_t k=0; s[k]!='\0'; k++){
		  if (s[k]!=' ') { // if not a blank
		  t[j]+=s[k];
		  j++;
		  }
		}
		t[j]='\0';

get rid of all the nonalpha and nonblank characters in a string d and get the new all lowercase string c		int j=0;
		for (int i = 0; d[i] != '\0'; i++) {
		  if (isalpha(d[i]) || d[i] == ' '){
		  c[j] = tolower(d[i]);
		  j++;
		  }
		}
		c[j] = '\0';

string(续)	C++ string	C string
get a substring	**s=s.substr(**POSITION,LENGTH**)**; 012345678 ***e.g.*** string s="dupli**cate**"; cout << s.substr(5,3); // cat	
add string in a certain position	**s=s.insert(**POSITION**,t) ;**	
clip off the first several characters of a string	**s=s.substr(**POSITION,**s.size()-**POSITION**);**	
clip off several consecutive characters in a string	**s=s.erase(**POSITION,LENGTH**);**	
erase the whole string	**s=s.erase();**	s[0]='\0';
input a string 1	**getline (cin, s);**	**cin.getline(s,**LENGTH**);**
input a string 2 (possibly wrong)	cin >> s; // wrong if CR LF is stored, need to add before: **cin.ignore(10000,'\0');**	~~cin >> s;~~ //wrong, only the first char is read in
output a string	cout << s;	cout << s;

(2) Definitions // s is string

	C++ string	C string
chars:	s[COLUMN]	s[LINE][COLUMN]
strings:	s	s[LINE]

(3) #include <cctype> // can be used in both C++ string and C string

a) return bool	b) change value
isalpha(s[POSITION**]);** **islower(s[**POSITION**]);** **isupper(s[**POSITION**]);** **isdigit(s[**POSITION**]);**	**tolower(s[**POSITION**]);** **toupper(s[**POSITION**]);**

2. array

(1) initialization/declaration: we cannot declare an array that has nothing in it.

```
int a1[3]={10, 20, 30}; //totally fine
int a2[]={10, 20, 30}; //compiler will count
int a3[3]; //3 uninitialized int values, need a[0]=…; a[1]=…; a[2]=…
int a4[]; // Error! Won't compile!
```

(2) set every element to the same value:

a) 0: we can use **int a[**LENGTH+1**]={0}**;

b) not 0: we can use **memset(a,** *'INT OR CHAR'*, *LENGTH_TO_SET***);**

(3) other notes:

a) syntax: <type> <name> [<array size>] ***e.g.*** int ar[10];

b) position starts from 0; // need length n+1 when wanting n elements in it

c) only exited int or const variable int can be used to declare an arry

```
// you can't use the read in number to declare an array!
int n=100;
cin >> n;
double a[n]; // WRONG! The value of n is not known at compile time.
```

d) No way just from the name of the array to know the length of it.

...dayInMonth.size()...; ...dayInMonth.length...// WRONG! Not in C++

// so the array size is often passed with it, but not required

e) multi-dimensional array need any dimension except the first, when compile

3. function

(1) return:

1) every branch need to return a value => don't forget to return default value

2) as soon as a function return the value, it goes back to main.

=> return in right position, i.e. in the right loop

e.g. bool contains(string s, char c) {

```
for (int k = 0; k != s.size(); k++) {
    if (s[k] == c) {return true};
// else {return false}; It would do if char is the first letter in string
} return false; } // as a default return, it should be outside of the loop
```

(2) pass value by reference

int &a=b; // set the reference of b to a. when using this in parameter in function,

// a and b are both the names for the thing in the memory,

// a and b change at the same time.

(3) other note:

a) syntax: typeOfReturningValue functionName (listOfParameters-Input)

b) when calling a function, not include the variable type

c) define a function outside another function

d) make sure variable types are right, or compiler will cast type with complexity

4. syntaxes to remember

(1) set precision

cout.setf(ios::fixed)

cout.precision(*INTEGER_NUMBER*);

(2) switch

switch (*INTEGER_NUMBER OR CHARACTER*){

```
case INTEGER1 OR CHARACTER1:
    ...;
  break;
case default:
    ...;}
```

(3) break and continue 针对最靠近的一个循环有效

(4) conditional operator

max = (n1 > n2) ? n1 : n2; ⇔ if (n1>n2) {max=n1}; else {max = n2};

5. type cast static_cast<TYPE_CHANGE_TO>(VARIABLE1)...;

B. USUAL MISTATES TO AVOID AND

1. Avoid Off-One Error, Think Thoroughly about '=': when conditions are complex, pay special attention to things related to <, >, <=, >= and plug in examples to see whether the assumption is right. ***e.g.*** for (int i=0; i<5; i++) *1,2,3,4,5* for (int i=0; i<=5; i++) *1,2,3,4,5,6* for (int i=1; i<5; i++) *1,2,3,4*

e.g. nonnegative: >=0; VS ! nonnegative: <0 辗转相除法的死循环包括 0 和负数

2. Always Adding '\0' in C String as Boundary, when writing and examining. Especially in while loop, don't forget this stay in loop condition. ***e.g.*** while (*some condition* && cleanDoc[iCount] != '\0') …

3. Use semicolon ; correctly in different loops

for	while	do-while
for (initialization; stay-in loop condition; pre-next-iteration) { statement; }	while (stay-in-loop condition) { expression; pre-next-iteration; }	do { expression; pre-next-iteration; }while (stay-in-loop condition);

4. Avoid Dangling Else Error: Write {} for every if, else if, else

5. Avoid No Initialization and Watch Out for Scoping

e.g. when using c string and want to add '\0' at the end of it, you have to declare the counter outside of the for or while loop. Otherwise, s[count] is undefined.

6. Avoid Overflow Error

int f=1000;

int g=f*f*f;

int h=f*g; //727379967 int∈[-2^{31}, 2^{31}-1] h 的值超过了 int 的最大值,溢出错误

7. Avoid Infinite Loop: use Boolean and unary operator correctly

```
int n=17;
while (n=1) { // this single = sign is wrong, it should be a double equal sigh ==
      cout << "n is one" << endl;
      n--;
} // this loop is infinite, as while(1) is an infinite loop
```

C. SPECIAL NOTES

1. 开始之前看清楚要求，题目上用不同记号标清楚

2. 带入数据人工编译，各个分段要齐全，要考虑到各种特殊情况

3. 在交卷之前，一定要再仔细核对一遍要求，不要提前交卷

if s = "", toupper[0] is undefined behavior

empty string--- c++: string s; <==> c: char s[100] = "";(不能用 char s[100] => uninitialized garbage value)