

登金陵鳳凰臺

鳳凰臺上鳳凰遊，
鳳去臺空江自流。
吳宮花草埋幽徑，
晉代衣冠成古邱。
三山半落青山外，
二水中分白鷺洲。
總為浮雲能蔽日，
長安不見使人愁。

～李白

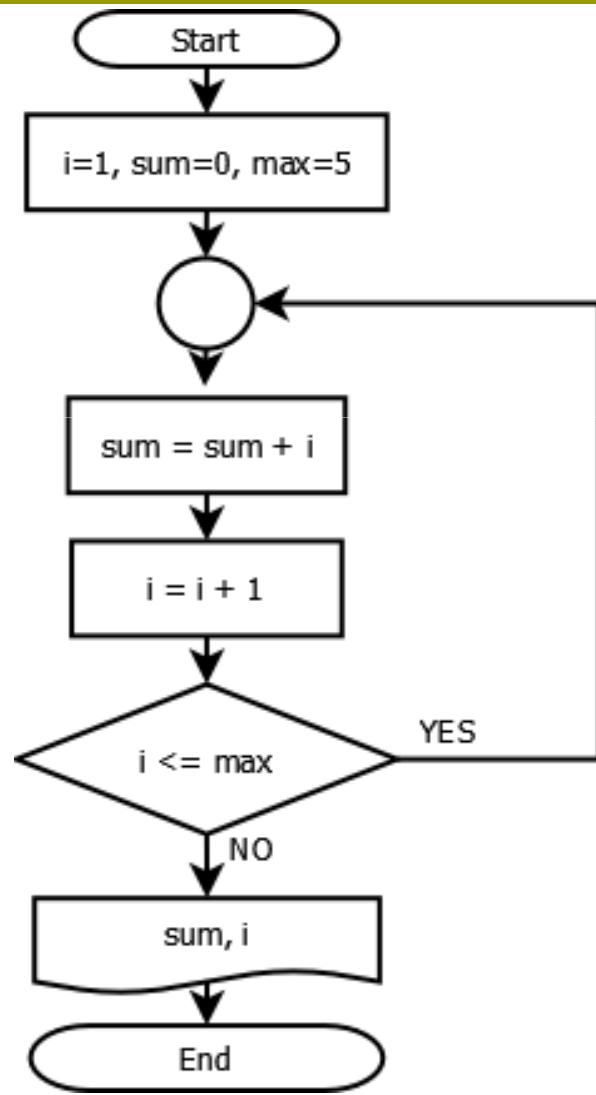


Unconditional Branching

```
myLabel: cout << "myLabel is here";  
.  
.  
.  
goto myLabel;
```

- ❑ Whenever possible, you should avoid using `gotos` in your program.

Loop (Ex3_07 in P.139)



Loop (Ex3_07 in P.139)

```
int i = 1, sum = 0;  
const int max = 5;
```

KevinLabel:

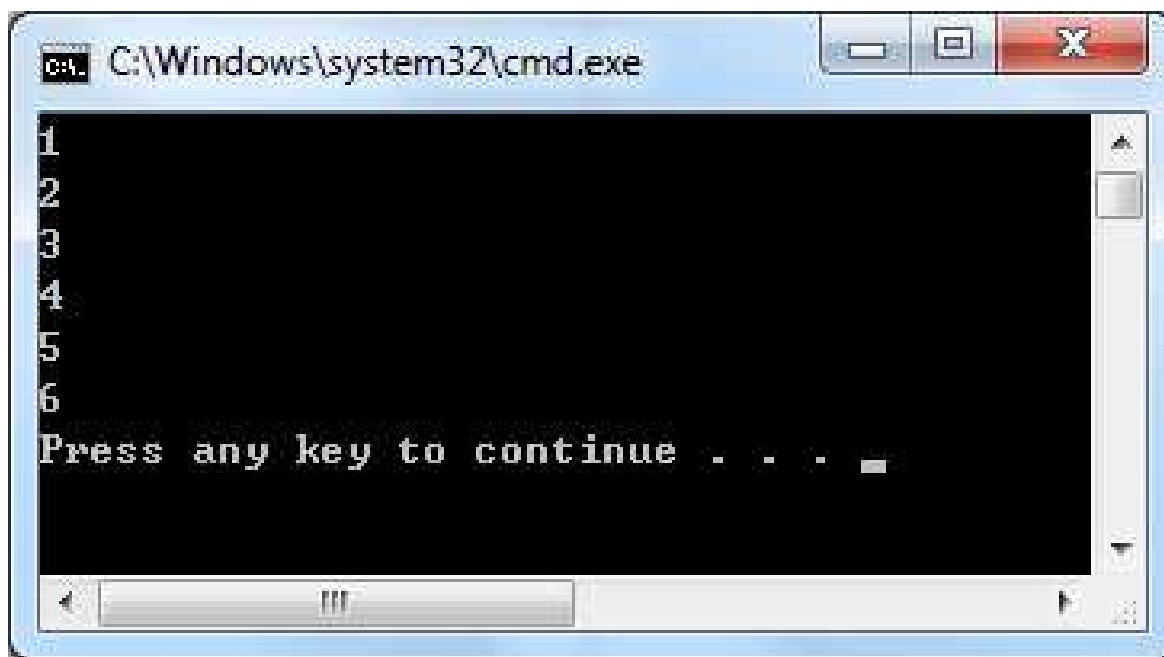
```
    sum += i;  
    if (++i <= max)  
        goto KevinLabel;
```

i = 4, sum = 10

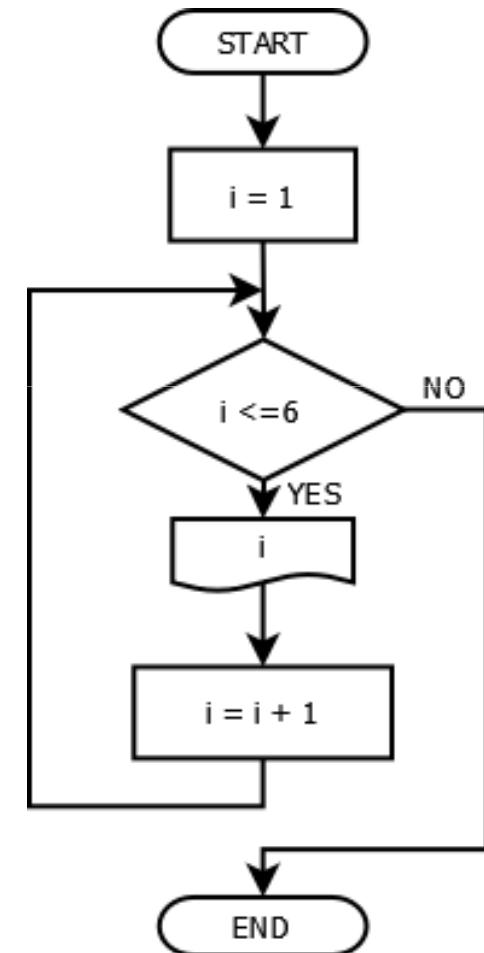
```
cout << "sum=" << sum << endl  
    << "i = " << i << endl;
```

The for Loop

```
for ( i=1; i<=6; i++ )  
    cout << i << endl;
```



A screenshot of a Windows command prompt window titled "cmd C:\Windows\system32\cmd.exe". The window displays the numbers 1 through 6, each on a new line, representing the output of the for loop. At the bottom of the window, the text "Press any key to continue . . ." is visible.



Using the for Loop for Summation

```
int i = 0, sum = 0;  
const int max = 5;  
  
for (i=1; i<=max; i++)  
    sum += i;           i = 5, sum = 00
```

- General form of the for loop:
 - `for (initializing_expression;
 test_expression; increment_expression)
 loop_statement;`

Nested for Loop

```
const int N = 5;  
  
int i, j;  
for (i=1; i<=N; i++)  
{  
    for (j=1; j<=i; j++)  
        cout << '*';  
    cout << endl;  
}
```

```
*  
* *  
* * *  
* * * *  
* * * * *
```

- ❑ A block of statements between braces could replace the single *loop_statement*.

Increment/Decrement of the Counter

```
for (i=1; i<=N; i++)
{
    for (j=1; j<=i; j++)
        cout << '*';
    cout << endl;
}

for (i=N; i>=1; i--)
{
    for (j=1; j<=i; j++)
        cout << '*';
    cout << endl;
}
```

```
*
```

```
**
```

```
***
```

```
****
```

```
*****
```

```
*****
```

```
***
```

```
**
```

```
*
```

ASCII Table

```
#include <iostream>
using std::cout;
using std::endl;

int main()
{
    unsigned char c;
    for (c=32; c<=126; c++)
    {
        if (c % 8 == 0) cout << endl;
        cout << c << '\t';
    }
    cout << endl;
    return 0;
}
```

The screenshot shows a Windows command-line interface (cmd.exe) window titled 'C:\Windows\system32\cmd.exe'. The window displays an ASCII table where each row contains two columns of characters. The first column consists of uppercase letters (A through Z), lowercase letters (a through z), and various punctuation marks and symbols. The second column consists of their corresponding lowercase counterparts and additional symbols. The table is organized into 8 rows and 8 columns. A message at the bottom of the window reads 'Press any key to continue . . .'. The window has standard Windows-style scroll bars on the right side.

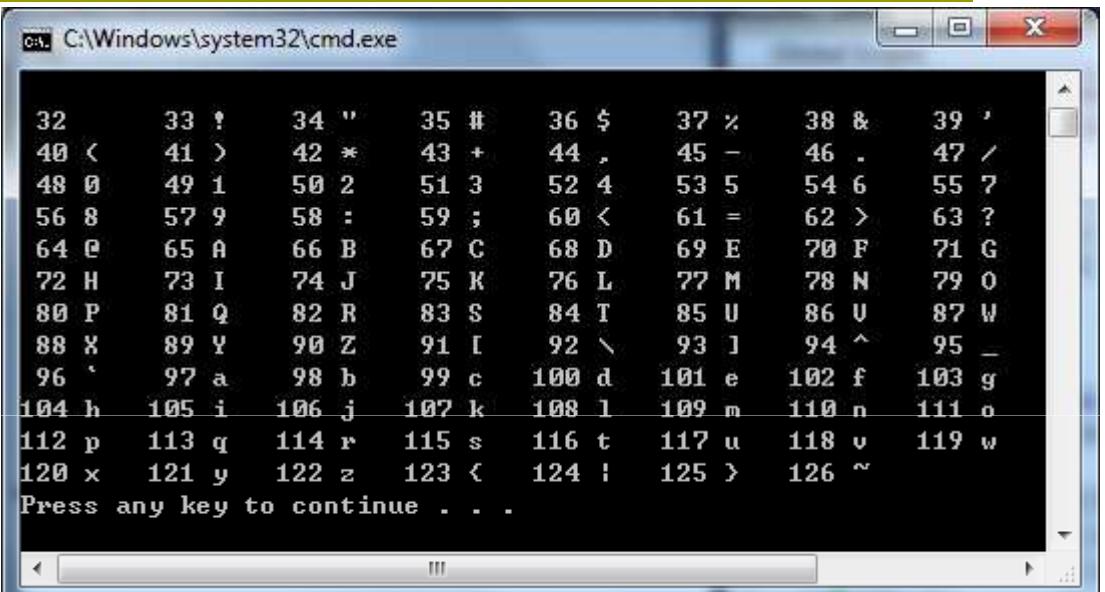
	!	"	#	\$	%	&	'
<	>	*	+	,	-	.	/
0	1	2	3	4	5	6	7
8	9	:	;	<	=	>	?
E	A	B	C	D	E	F	G
H	I	J	K	L	M	N	O
P	Q	R	S	T	U	U	W
X	Y	Z	[\]	^	_
.	a	b	c	d	e	f	g
h	i	j	k	l	m	n	o
p	q	r	s	t	u	v	w
x	y	z	{	:	}	~	

ASCII Table (2)

```
#include <iostream>
#include <iomanip>

using std::cout;
using std::endl;
using std::setw;

int main()
{
    unsigned char c;
    for (c=32; c<=126; c++)
    {
        if (c % 8 == 0) cout << endl;
        cout << setw(3) << static_cast<int>(c) << ' ';
        cout << c << '\t';
    }
    cout << endl;
    return 0;
}
```



The screenshot shows a Windows command-line interface (cmd.exe) window titled 'C:\Windows\system32\cmd.exe'. The window displays the ASCII table, listing characters from 32 to 126. The characters are grouped by their ASCII values, with each group containing a decimal value, a standard character representation, and a corresponding escape sequence. The table is organized into 8 columns. The last line of the output is 'Press any key to continue . . .'.

32	33 !	34 "	35 #	36 \$	37 %	38 &	39 '
40 <	41 >	42 *	43 +	44 ,	45 -	46 .	47 /
48 0	49 1	50 2	51 3	52 4	53 5	54 6	55 7
56 8	57 9	58 :	59 ;	60 <	61 =	62 >	63 ?
64 @	65 A	66 B	67 C	68 D	69 E	70 F	71 G
72 H	73 I	74 J	75 K	76 L	77 M	78 N	79 O
80 P	81 Q	82 R	83 S	84 T	85 U	86 V	87 W
88 X	89 Y	90 Z	91 [92 \	93]	94 ^	95 _
96 `	97 a	98 b	99 c	100 d	101 e	102 f	103 g
104 h	105 i	106 j	107 k	108 l	109 m	110 n	111 o
112 p	113 q	114 r	115 s	116 t	117 u	118 v	119 w
120 x	121 y	122 z	123 <	124 >	125 ~		

Variation on the for Loop

- ❑ Declare the counter i within the loop scope.
The loop statement can be empty.
 - ```
for (int i = 1; i<=max; sum+= i++)
 ;
```
- ❑ You can omit the initialization expression
  - ```
int i = 1;  
for (; i <= max; i++)  
    sum += i;
```
- ❑ Use the comma operator to specify several expressions:
 - ```
for (i=0, power=1; i<=max; i++, power *=2)
```

# Summing Up Odd Numbers

---

```
#include <iostream>
using std::cout;
using std::endl;

int main()
{
 int i;
 int sum=0;
 for (i=1; i<=9; i+=2)
 sum += i;
 cout << sum << endl;
 return 0;
}
```

# Prime Number Test

---

```
#include <iostream>

using std::cin;
using std::cout;
using std::endl;

int main()
{
 int n;
 int isPrime = true;
 cin >> n;

 if (n % 2 == 0) isPrime = false;
 for (int i=3; i<n; i+=2)
 if (n % i == 0) isPrime = false;
 if (isPrime)
 cout << n << " is a prime number." << endl;
 else
 cout << n << " is NOT a prime number." << endl;
 return 0;
}
```

## break vs. continue

---

- The keyword `continue` allows you to skip the remainder of the current iteration in a loop and go straight to the next iteration.
- The keyword `break` provides an immediate exit from a loop.
  
- (See P.145 and P.146)

# Other Types of Loop

---

- The while loop
  - `while (condition)  
loop_statement;`
  - Ex3\_12.cpp on P.151
  
- The do-while Loop
  - `do  
{  
loop_statements;  
} while (condition);`
  - Always executed at least once.
  
- You may see infinite loops like
  - `while (true)  
{  
...  
}  
while (1)  
{  
...  
}  
for (;;) {  
...  
}`

# Greatest Common Divisor

---

```
#include <iostream>

using std::cin;
using std::cout;
using std::endl;

int main()
{
 int a, b, temp;
 cout << "a=? ";
 cin >> a;
 cout << "b=? ";
 cin >> b;
 if (a==0 && b==0)
 {
 cout << "I don't know how to calculate their gcd.\n";
 return 1;
 }
 cout << "The greatest common divisor of " << a << " and " << b << " is ";
 while (b != 0)
 {
 a %= b;
 temp = b; b = a; a = temp; // swap a,b
 }
 cout << a << endl;
 return 0;
}
```



# Exercise

---

- P.164
  - Exercise 1, 2
- You don't need to upload, but we shall have a quiz at the end of this class.
- Also try to run the sample code introduced in this chapter, to get a feeling about the decisions and loops of C++ language.

# Homework

---

## ❑ Prime number <= N

- Extend the “Prime Number Test” program to list all prime numbers less than or equal to N, where N is input from the user.

## ❑ Factorization

- Input N, and factorize N.
- For example,  $12 = 2 * 2 * 3$

## ❑ Least Common Multiple

- Input a, b, and output  $\text{lcm}(a,b)$ .
- For example,  $\text{lcm}(12,8)=24$

# Homework (bonus)

---

- Perfect Number
  - In [number theory](#), a **perfect number** is a [positive integer](#) that is equal to the sum of its proper positive [divisors](#); that is, the sum of its positive divisors excluding the number itself.
- For example,
  - $6 = 1 + 2 + 3$
  - $28 = 1 + 2 + 4 + 7 + 14$
- Write a program to list all perfect numbers less than or equal to N, where N is input from the user.