Chapter 3 Review

- Find errors:

A. #INCLUDE <stdio.h>

B. file myfile;

C. *myfile = fopen (C3_1.DAT, r);

D. fscanf (“myfile”, “%4d %5d\n”, WEEK, YEAR);

E. close (“myfile”);

Chapter 3 Review

- Fill in the blanks:

Type	Bytes	Format type for printf()
char		
int		
long		
float		
double		

Chapter 3 Review

- Q & A:

A. How to calculate the square root of a variable x?

B. How to calculate x^y?

C. Do we need a header file?

D. What's the difference between scanf() and getchar()?

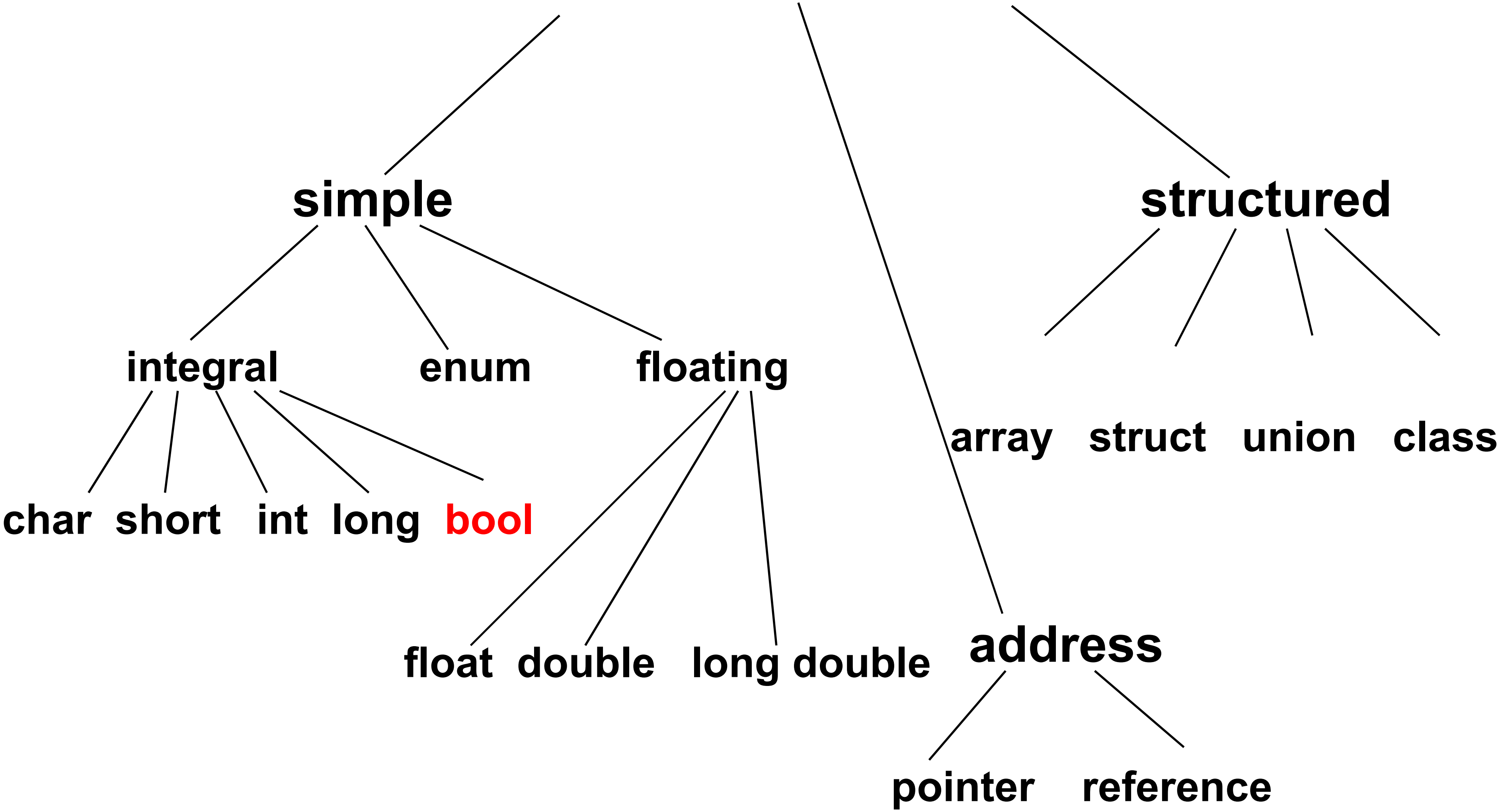
Chapter 4

Beginning Decision Making and Looping

Flow Control

- Relational expressions
- ***if***, ***if-else*** statement
- Looping statements
  - ***while***
  - ***do while***
  - ***for***

C++ Data Types



simple

structured

integral

enum

floating

char short int long bool

float double long double

array struct union class

address

pointer reference

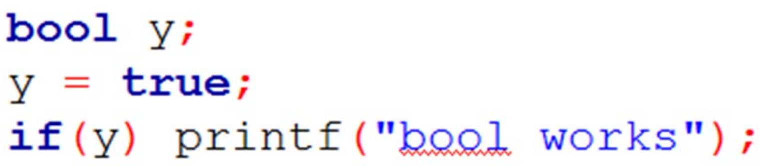
The ***bool*** Data Type

The ***bool*** data type is a built-in type consisting of just two values, the constants ***true*** and ***false***.

false	0
true	ANY non-zero value represents true

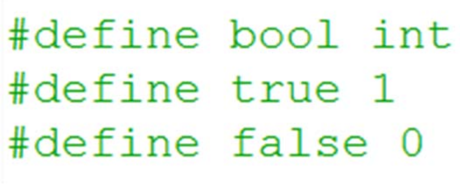
C data types

- C does **NOT** have ***bool*** until C99 by including **<stdbool.h>**
- Try it! E.g.:



```
bool y;
y = true;
if(y) printf("bool works");
```

- Other ways to define and use ***bool*** type in ANSI C?



```
#define bool int
#define true 1
#define false 0
```

4.1 *if Control Structure and Relational Expressions*(Page 125)

- A relational expression compares two values
- General format

  operand relational_operator operand

  Mark < 60

- Produces a result of either ***true*** or ***false***
- **Operand** can be variables, or any arithmetic expression.

  i + j > 3 * k

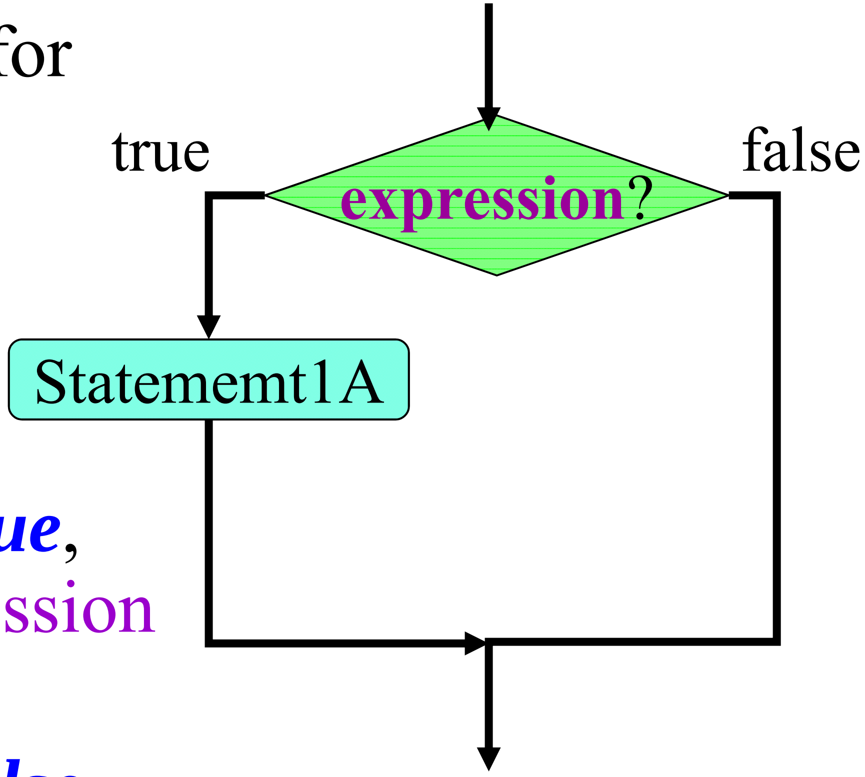
Relational Operator	Meaning
<	Less than
<=	Less than or equal to
==	Equal to
>	Greater than
>=	Greater than or equal to
!=	Not equal to

if* *Statement (Page 127)

- *if* statement often is used for decision making
- General form

```
if (expression)
    statement;
```

- If **logical expression** is ***true***, **statement** behind ***if*** expression is executed
- If **logical expression** is ***false***, **statement** behind ***if*** expression is **not** executed



if Statement

```
#include "stdafx.h"
#include <stdlib.h>
#include <time.h>
int _tmain(int argc, _TCHAR* argv[])
{// C4_1_If Statement
int nRandNum,nGuessNum,nMin,nMax;
  srand((unsigned)time(NULL));
  nMin=1;nMax=2;
  nRandNum=(float)rand()/RAND_MAX*(nMax-nMin)+nMin;
  scanf("%d",&nGuessNum);
  if(nGuessNum==nRandNum)
    printf("nRandNum=%d.You are lucky\n",nRandNum);
  return 0;
}
```

Compound Statement

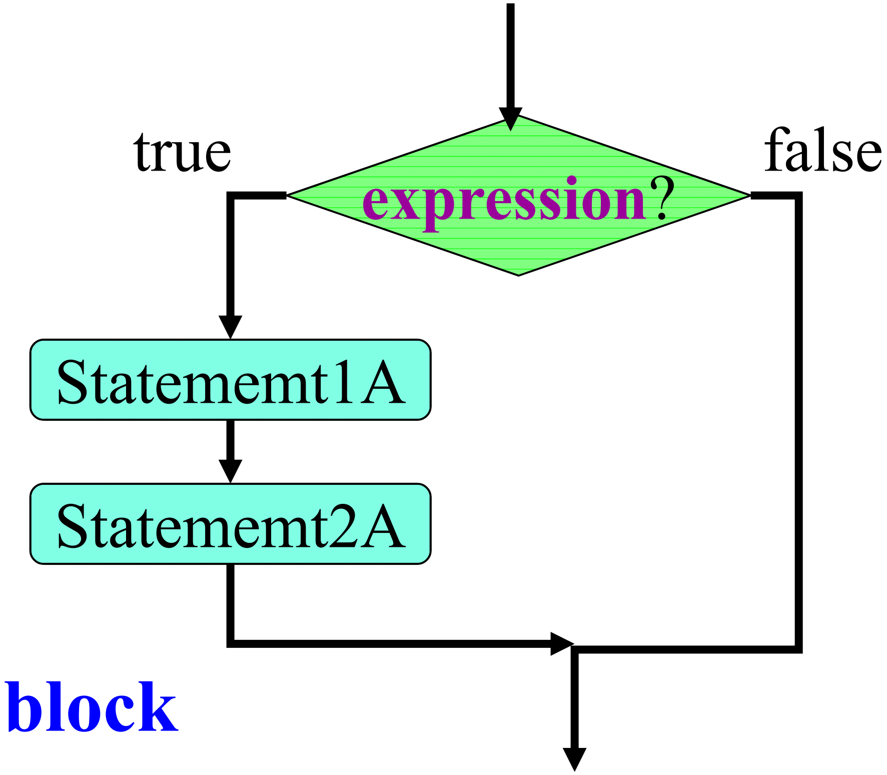
- A compound statement is any number of statements contained between braces
- Although only a single statement is permitted in both the if and else parts of ***if-else*** statement, this statement can be a single compound statement

```
{
   z=x+y;
   t=z/100;
}
```

Block if(Page 127)

- A block ***if*** statement

```
 if (expression)
 {
       statements
       …
 }
```



- Several statements in the **block** (between two braces) are executed
- Otherwise the entire block is ignored

Block

```
int nRandNum,nGuessNum,nMin,nMax,nLuckyCounter=0;
  srand((unsigned)time(NULL));
  nMin=1;nMax=2;
  nRandNum=(float)rand()/RAND_MAX*(nMax-nMin)+nMin;
  scanf("%d",&nGuessNum);
  if(nGuessNum==nRandNum)
  {
    printf("nRandNum=%d.You are lucky\n",nRandNum);
    nLuckyCounter++;
  }
```

Equality Operator (Page 128)

- Difference between = and = =

  = is assignment but = = is a relational operator

  must not be confused as will cause serious errors

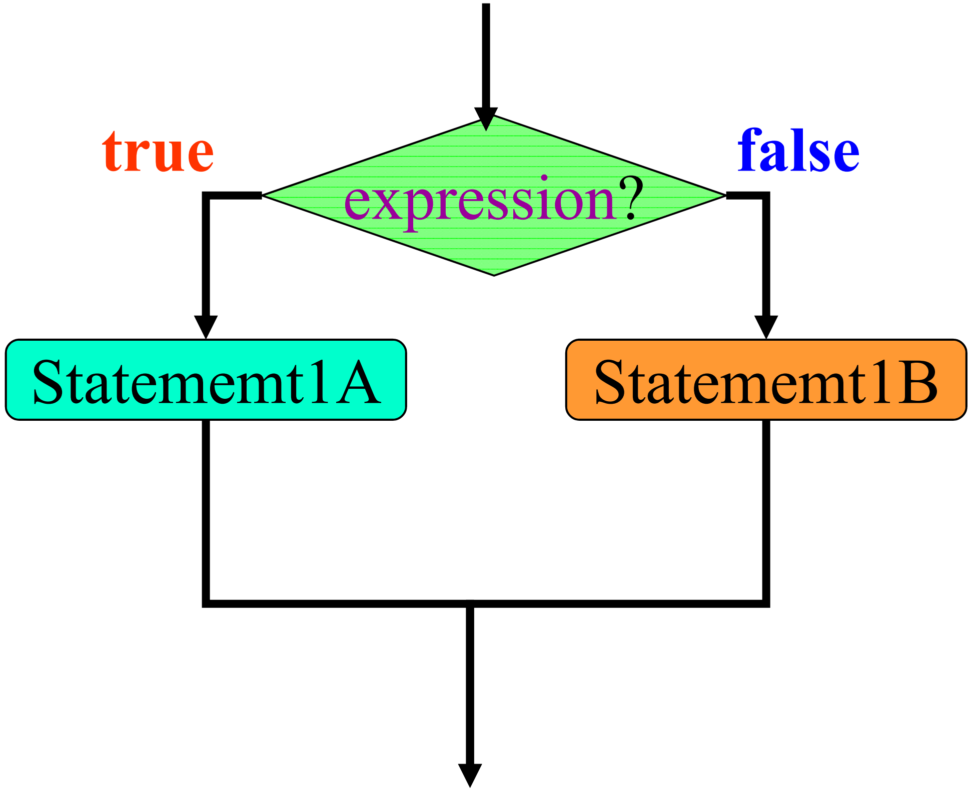
Equality Operator (Page 129)

- Not recommended to use= = to compare values of ***double*** or ***float***!
  - Typical C compiler carries many significant digits for the real variable types
  - If two fractional values are compared with = = and they differ in only the last significant digit, then the result of a comparison with = = is false
- In most cases, we are interested only if the numbers are nearly or very nearly equal, e.g.,
  x = 0.1;
  x * 9 = = 0.9 will return False
- So how do we compare two real values?
  - use the ***fabs*** function and <
  - to compare the double values of a and b
    if (fabs (a-b) < 1.0e-10)
  - 1.0e-10 assume using double,
  - you will need to decide how small that number should be

4.2 *if-else* Control (Page 130)

- *if-else* syntax

```
if (expression)
{
   executable statement 1A;
    ...
}
else
{
   executable statement 1B;
   ...
}
```



- If the **expression** is **true**, statements in the **Statememt1A** block are executed
- If the **expression** is **false**, control is transferred to the false block **Statememt1B**

if-else Statement

```
int nRandNum,nGuessNum,nMin,nMax,nLuckyCounter=0;
  srand((unsigned)time(NULL));
  nMin=1;nMax=2;
  nRandNum=(float)rand()/(RAND_MAX+1)*(nMax-nMin+1)+nMin;
  scanf("%d",&nGuessNum);
  if(nGuessNum==nRandNum)
  {
    printf("nRandNum=%d.You are lucky\n",nRandNum);
    nLuckyCounter++;
  }
  else
    printf("nRandNum=%d.You are unlucky\n",nRandNum);
```

Try!

Input 2 numbers, a and b, then print out these 2 numbers in order(from small to large).

```
#include "stdafx.h"
int _tmain(int argc, _TCHAR* argv[])
{// C4_1_If Statement
int    a,b;
  printf("a=");
  scanf("%d",&a);
  printf("\nb=");
  scanf("%d",&b);
  …
  printf("\nThe larger=%d, the smaller=%d\n",a,b);
  return 0;
}
```

Try!

```
#include "stdafx.h"
int _tmain(int argc, _TCHAR* argv[])
{// C4_1_If Statement
int    a,b,t;
  printf("a=");
  scanf("%d",&a);
  printf("\nb=");
  scanf("%d",&b);
  if(a>b)
  {
     t=a;
     a=b;
     b=t;
  }
  printf("\nThe larger=%d, the smaller=%d\n",a,b);
 return 0;
}
```

Conditional Operator (Page 131)

? :

conditional operator

- Requires three operands
- Format

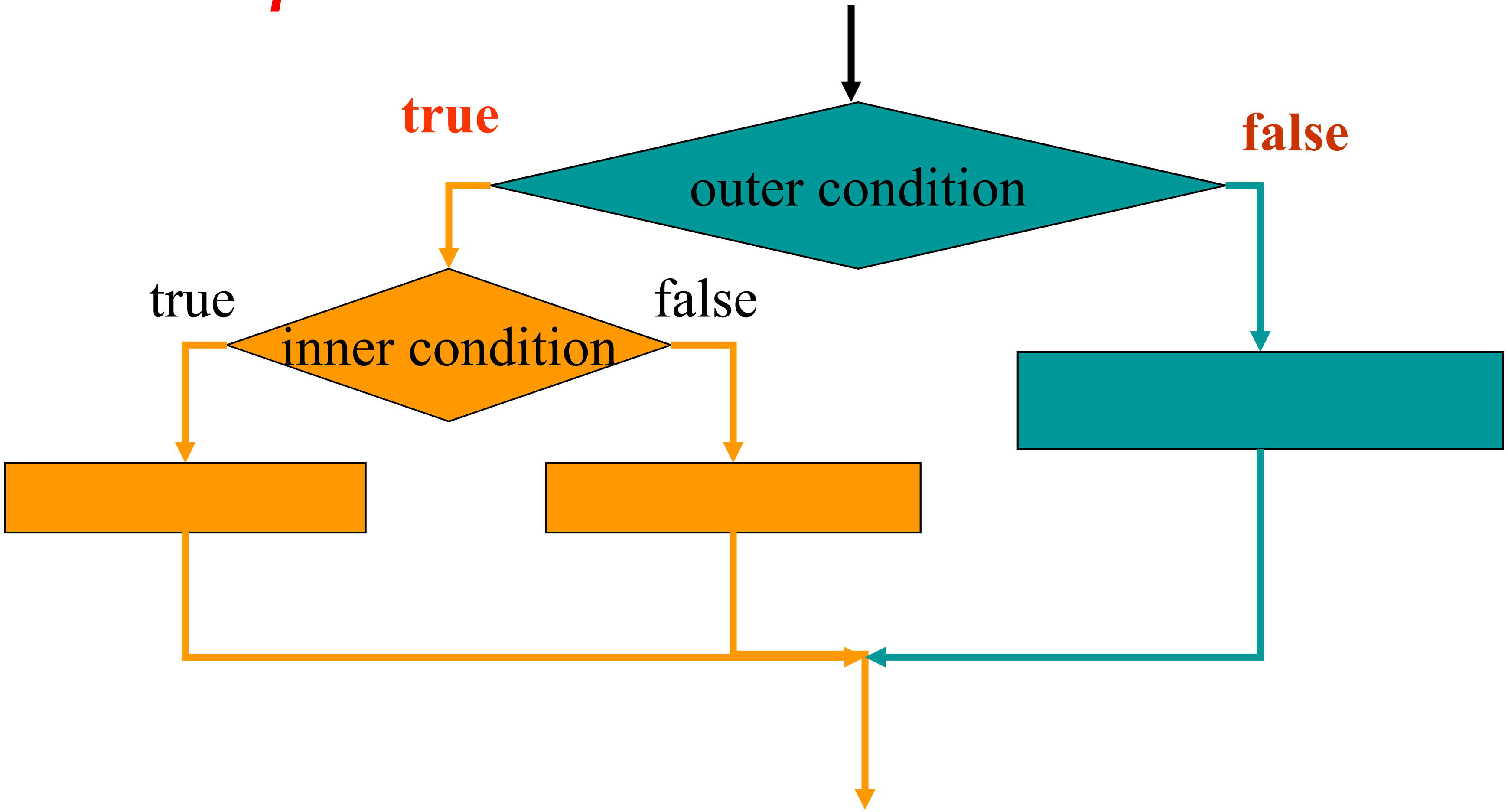
```
expr1?expr2:expr3
(y<z) ?   y   :   z;
```

- If **expr1** is true, **expr2** is evaluated
- If **expr1** is false, **expr3** is evaluated
- x = (y<z) ? y : z;

assigns x the value of the smaller of y and z

4.3 *Nested* ***if-else*** (Page 135)

- An ***if-else*** control structure can be contained within another ***if-else***

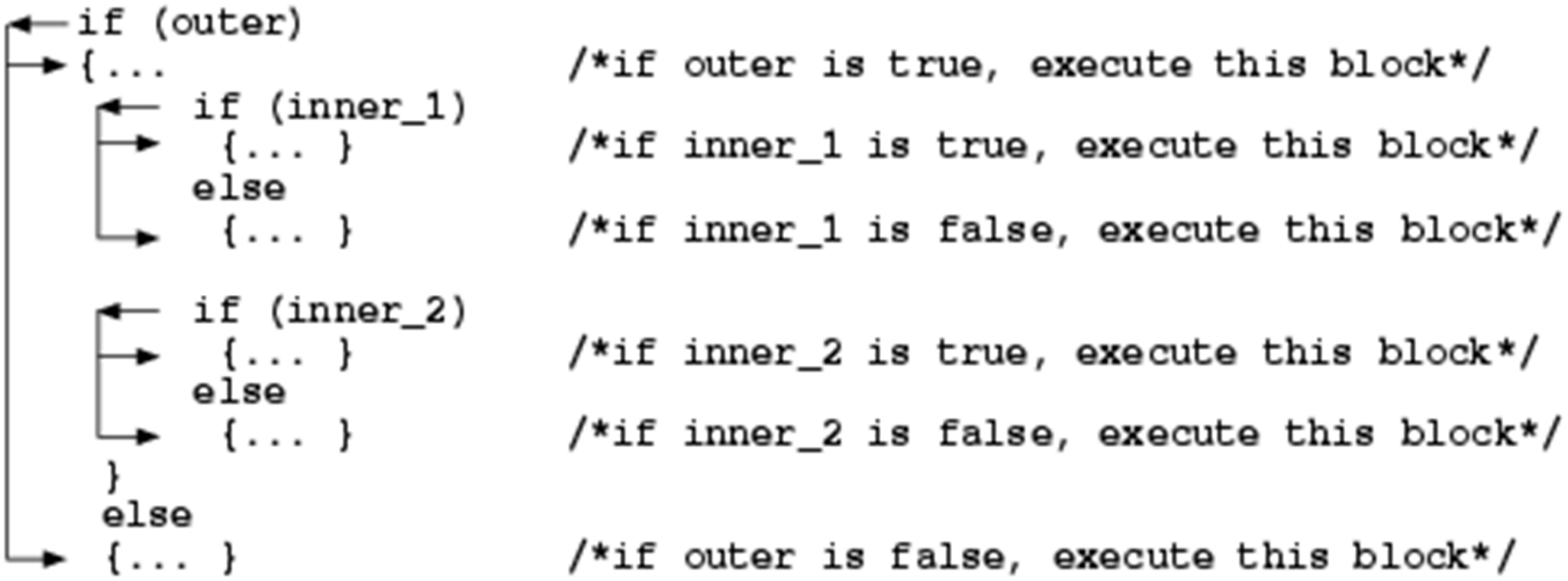


Nested *if-else*

```
int nRandNum,nGuessNum,nMin,nMax,nLuckyCounter=0;
 srand((unsigned)time(NULL));
 nMin=1;nMax=2;
 nRandNum=(float)rand()/(RAND_MAX+1)*(nMax-nMin+1)+nMin;
 scanf("%d",&nGuessNum);
 if(nGuessNum==nRandNum)
 {
   printf("nRandNum=%d.You hit! Try again\n",nRandNum);
   nRandNum=(float)rand()/(RAND_MAX+1)*(nMax-nMin+1)+nMin;
   scanf("%d",&nGuessNum);
   if(nGuessNum==nRandNum)
     printf("nRandNum=%d.You are very lucky\n",nRandNum);
   nLuckyCounter++;
 }
 else
   printf("nRandNum=%d.You are unlucky\n",nRandNum);
```

Nested if-else

- Use indentation – indent each pair of if-else so that inner else is paired with inner if and outer else is paired with outer if
- An else clause is associated with the closest previous if statement that has no other else statement



```
if (outer)
{...                    /*if outer is true, execute this block*/
    if (inner_1)
      {... }            /*if inner_1 is true, execute this block*/
    else
      {... }            /*if inner_1 is false, execute this block*/

    if (inner_2)
      {... }            /*if inner_2 is true, execute this block*/
    else
      {... }            /*if inner_2 is false, execute this block*/
}
else
{... }                  /*if outer is false, execute this block*/
```

Home Work

- Page 138-2 Write a program to input these ten incomes from a file

4.4 *Logical Expressions* (Page 135)

- Used to connect two relational expressions, e.g.,

  if **(x == 0 && y == 0)**

- Logical operator

  && for “and”

  || for "or“

  ! for "not“ - reverse logical value

A	B	A&&B	A||B	!A	!B
True	True	True	True	False	False
True	False	False	True	False	True
False	True	False	True	True	False
False	False	False	False	True	True

Try!

(1)Man retire after 60, lady retire after 50.

(2)Man retire after 60, intellectual lady retire after 55, labour lady retire after 50.

Input gender and age, then print out retirement status.

Try!

(1)Man retire after 60, lady retire after 50.

(2)Man retire after 60, intellectual lady retire after 55, labour lady retire after 50.

```
int       nAge;
char    cGender;
  scanf("%d",&nAge);
  scanf("%c",&cGender);
  if(cGender=='M' && nAge>=60)
     printf("Retired\n");
  if(cGender=='F' && nAge>=50)
     printf("Retired\n");
```

Question: How to print out working status?

Home Work

- Page 141-1 Given x = 200 and y= -400 , determine whether…
- Page 141-3 Suppose the yearly demand…

4.5 Order of Precedence (Page 142)

assuming a=4, b=22, and c= 0

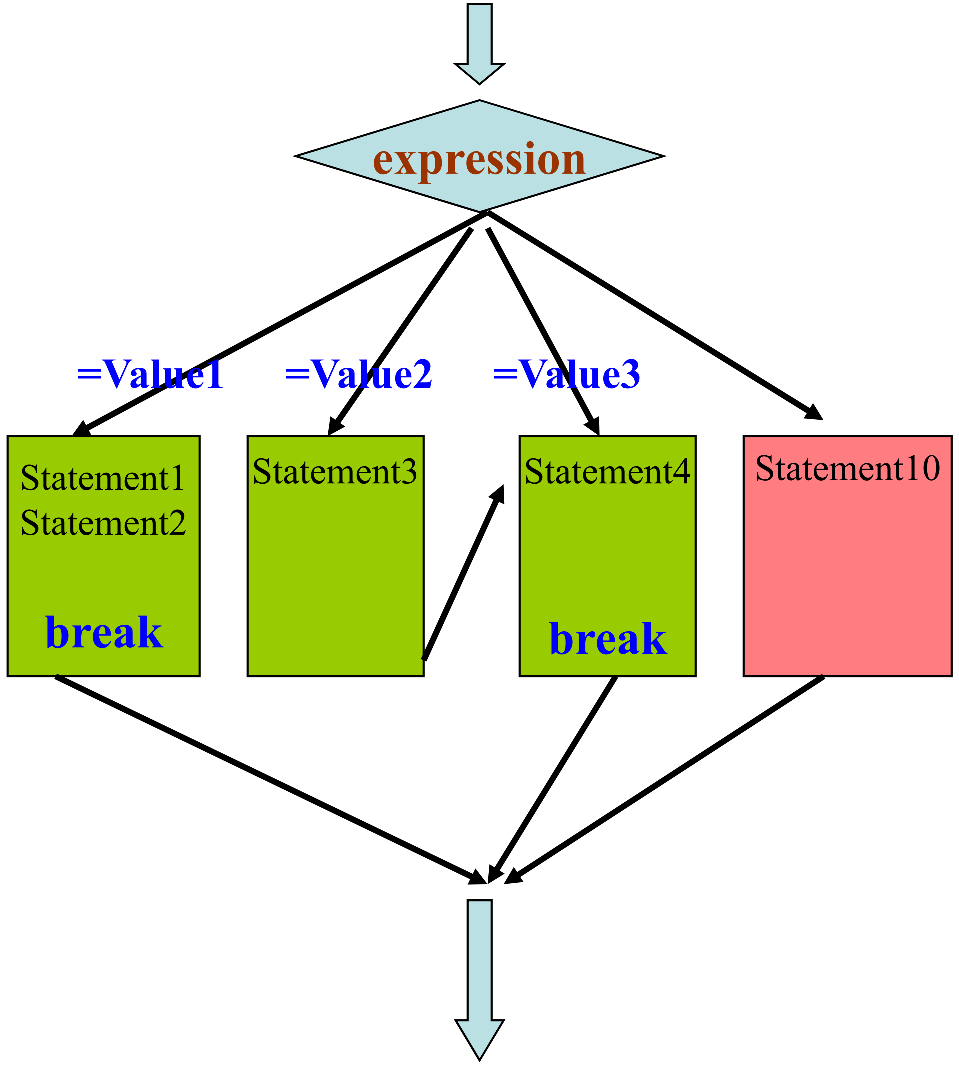
x =(a>b || b>c && a==b)

Operator	Name	Associativity	Precedence
(,)	Parentheses	L to R	1(highest)
++,--	Post-increment	L to R	2
++,--	Pre-increment	R to L	2
!	Logical NOT	L to R	3
+, -	Positive, negative sign	L to R	3
+=,-=,*=,/=,%=	Compound assignment	R to L	3
*, /	Multiplication, Division	L to R	4
+, -	Addition, Subtraction	L to R	5
= =,>=,<=,>,<,!=	Relational Operator	L to R	6
&&	Logical AND	L to R	7
||	Logical OR	L to R	8
=	Assignment	R to L	9 (Lowest)

4.6 *switch* *Statement* (Page 147)

```
switch(expression)
 {
   case Value1 : Statement1;
                 Statement2;
                 break;
   case Value2 : Statement3;
   case Value3 : Statement4;
                 break;
                   .

   default :     Statement10;
}
```



4.6 *switch* Statement (Page 147)

```
switch(expression)
 {
   case Value1 : Statement1;
                 Statement2;
                 break;
   case Value2 : Statement3;
   case Value3 : Statement4;
                 break;
                  .

   default :     Statement10;
}
```

- Looks for a match between the switch **expression** and the ***case*** label
- ***case*** label: constant followed by colon
- ***break*** statement send control to the point of the closing brace of ***switch***
- If no ***break*** statement is used, then the statements in the next ***case*** are executed
- ***default*** is a special label in the event that ***none*** of the case label constants agrees, control passes to the ***default*** labeled statement sequence
- ***switch*** control structures can be nested also

Count Random Number

```
int _tmain(int argc, _TCHAR* argv[])
{
int        nRandom,nMin,nMax;
int        nCounter1=0,nCounter2=0,nCounter3=0;
  srand((unsigned)time(NULL));
  nMin=1;nMax=3;
  nRandom=(float)rand()/RAND_MAX*(nMax-nMin+1)+nMin;
  switch(nRandom)
  {
     case 1:   nCounter1++;
               break;
     case 2:   nCounter2++;
               break;
     case 3:   nCounter3++;
               break;
     default: printf("Error Random Number\n");
  }
  return 0;
}
```

```
nMin=0;nMax=99;
nRandom=(float)rand()/RAND_MAX*(nMax-nMin+1)+nMin;
nRandom=nRandom/10;
switch(nRandom)
{
case 0:
case 1:
case 2:
case 3:
case 4:    printf("A_");
           nCounter1++;
           break;
case 5:
case 6:
case 7:    printf("B_");
           nCounter2++;
           break;
case 8:
case 9:    printf("C_");
           nCounter3++;
           break;
}
```

Home Work

- Page 155-3 Use a switch statement to write a program...

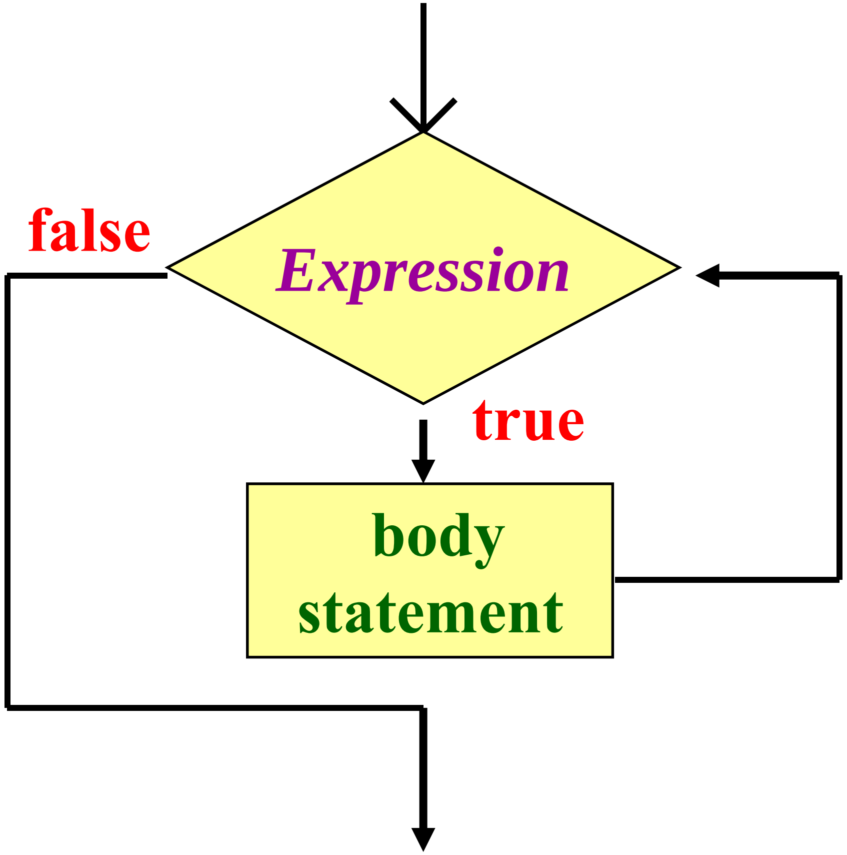
4.7 *while* Loop (Page 156)

- Iterative control structures involves the repeated execution of one or more statements

  while **(expression)**

  statement

- If result of **expression** is ***true***, statement is executed and expression will be tested again
- Leave the loop once expression is ***false***



Try!

$$nSum = \sum_{i=1}^{100} i = 1 + 2 + 3 + ... + 100$$

Try!

$$nSum=\sum_{i=1}^{100} i = 1+2+3+\ldots+100$$

```
int       i,nSum;
  nSum=0;
  i=1;
  while(i<=100)
  {
    nSum=nSum+i;
    i=i+1;
  }
```

Try: 1*2*...*?>=100

```
int _tmain(int argc, _TCHAR* argv[])
{
int     i,nProduct;
  nProduct=1;
  i=0;
  while(i<100)
  {
    i++;
    nProduct=nProduct*i;
    if(nProduct>=100)
      break;
  }
  printf("1*2*...*%d<100",i-1);
}
```

Try!

$$nSum = \sum_{i=1}^{100} i = 1 + 2 + 3 + \ldots + 100$$

$$nSum = \sum_{i=1}^{100} 3 * i = 3 + 6 + 9 + \ldots + 300$$

Try!

$$nSum=\sum_{i=1}^{100}3*i=3+6+9+...+300$$

```
int       i,nSum;
 nSum=0;
 i=1;
 while(i<=100)
 {
   nSum=nSum+3*i;
   i=i+1;
 }
```

Try!

$$nSum = \sum_{i=1}^{100} i = 1+2+3+\ldots+100$$

$$nSum = \sum_{i=1}^{100} 3*i = 3+6+9+\ldots+300$$

$$nSum = \sum_{i=1}^{100} \frac{(-1)^i}{3*i} = -\frac{1}{3}+\frac{1}{6}-\ldots+\frac{1}{300}$$

Try!

$$nSum = \sum_{i=1}^{100} \frac{(-1)^i}{3*i} = -\frac{1}{3} + \frac{1}{6} - \ldots + \frac{1}{300}$$

```
int      i,nSign;
float    fSum;
  fSum=0.0;
  nSign=-1;
  i=1;
  while(i<=100)
  {
    fSum+=nSign*1.0/(3*i);
    nSign=nSign*(-1);
    i=i+1;
  }
  printf("fSum=%f",fSum);
```

Generate Random Numbers

```
#include <stdlib.h>
#include <time.h>
int _tmain(int argc, _TCHAR* argv[])
{
int      nRandom,nMin,nMax,nNum;
  srand((unsigned)time(NULL));
  nMin=1;nMax=3;
  nNum=0;
  while(nNum<10)
  {
     nRandom=(float)rand()/RAND_MAX*(nMax-nMin+1)+nMin;
     printf("%d\n",nRandom);
     nNum++;
  }
  return 0;
}
```

Try!

Generate 100 random numbers from 1 to 3 and count their numbers

```
int   nRandom,nMin,nMax,nNum,nCounter1=0,nCounter2=0,nCounter3=0;
  srand((unsigned)time(NULL));
  nMin=1;nMax=3;
  nNum=0;
  while(nNum<100)
  {
     nRandom=(float)rand()/RAND_MAX*(nMax-nMin+1)+nMin;
     switch(nRandom)
     {
     case 1:nCounter1++;
              break;
     case 2:nCounter2++;
              break;
     case 3:nCounter3++;
              break;
     }
      nNum++;
  }
  printf("Counter1=%d\n",nCounter1);
  printf("Counter2=%d\n",nCounter2);
  printf("Counter3=%d\n",nCounter3);
```

Sentinels

- Data values used to signal either the start or end of a data series are called sentinels.
- The sentinel values must, of course, be selected so as not to conflict with legitimate data values.

Get π!

$$\frac{\pi}{4} = \sum_{i=1}^{\infty} (-1)^{i-1} \frac{1}{2*i-1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots$$

- Calculate approximate π until the absolute value of the last item is less than or equal to a given value(sentinel)

Get π!

$$\frac{\pi}{4}=\sum_{i=1}^{\infty}(-1)^{i-1}\frac{1}{2*i-1}=1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\ldots$$

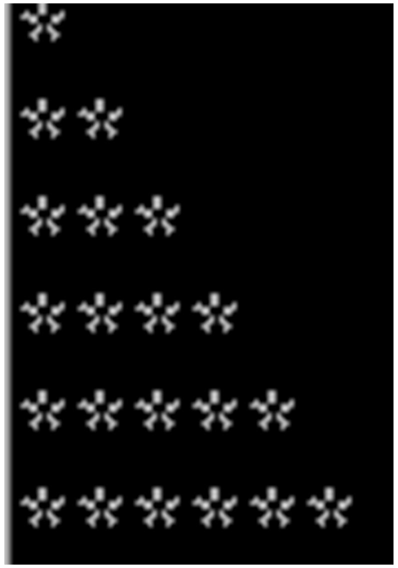
```
double   dSum,dItem;
long     i;
int      nSign;
  nSign=1;    i=1;    dSum=0.0;    dItem=1;
  while(fabs(dItem)>1e-5)
  {
     dSum+=dItem;
     nSign*=-1;
     i++;
     dItem=nSign/double(2*i-1);
  }
  dSum*=4;
  printf("Pi=%.20f\n",dSum);
```

Home Work

$$\ln(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \ldots$$

- **Get ln(1.1)=0.09531017980432486**
- **1+2+...+?>10000**

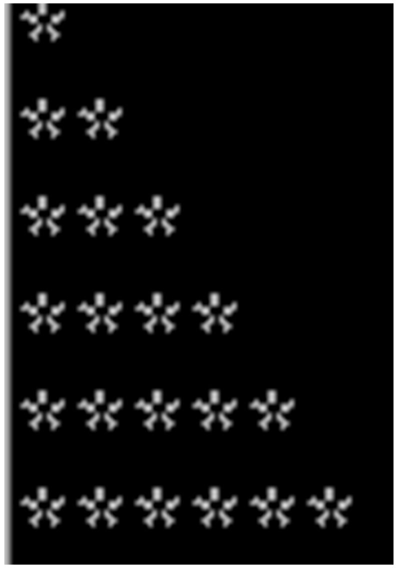
Try!



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

Try!

```
int     i,j;
   i=1;
   while(i<=6)
   {
     j=1;
     while(j<=i)
     {
        printf("*");
        j++;
     }
     printf("\n");
     i++;
   }
```

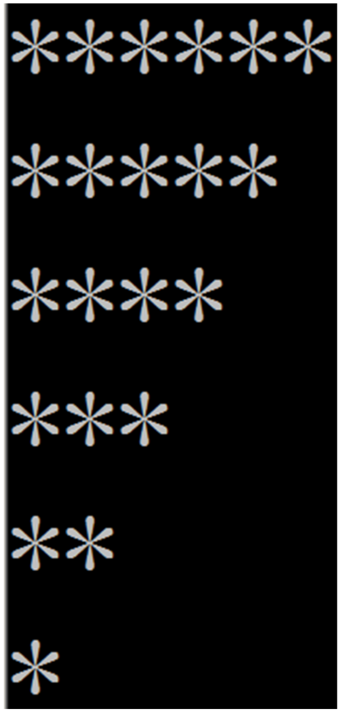


```
*
**
***
****
*****
******
```

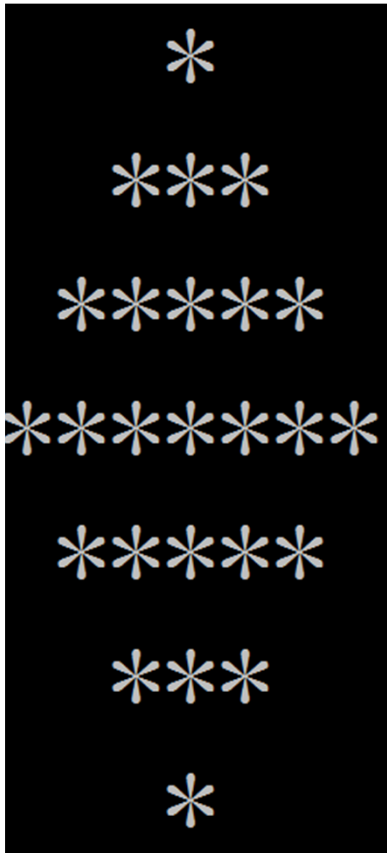
Home Work

- Page 159-3 Use a while loop to show that...
- Page 159-4 Use if and other C statements to determine...
- Page 162-2 Use a while loop to calculate the factorial....

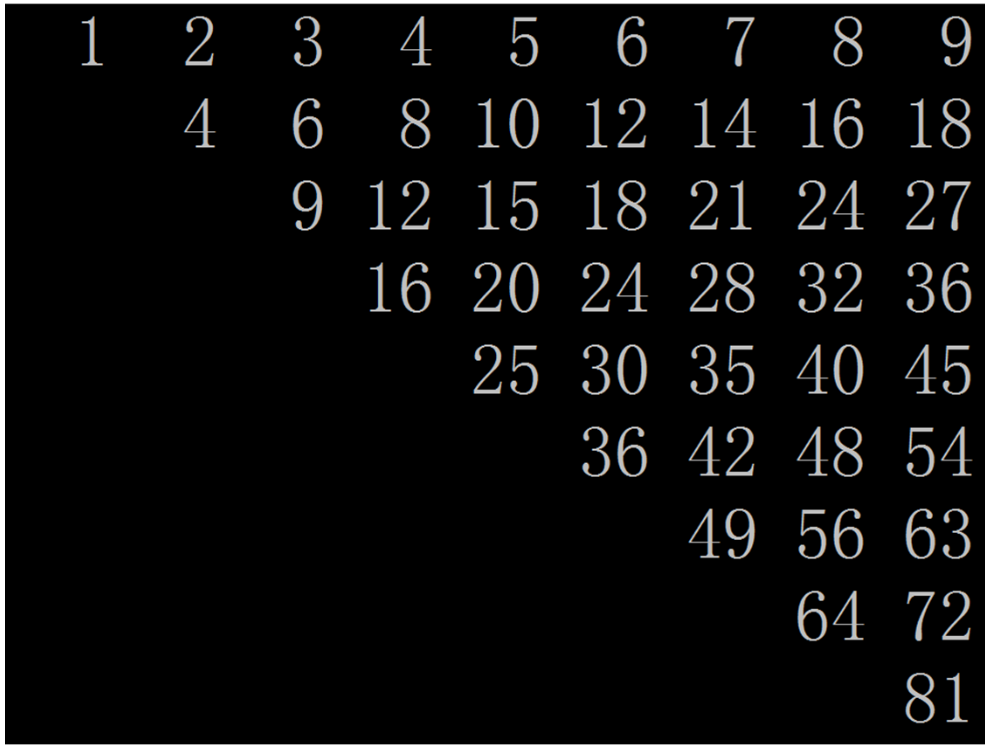
Home Work



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

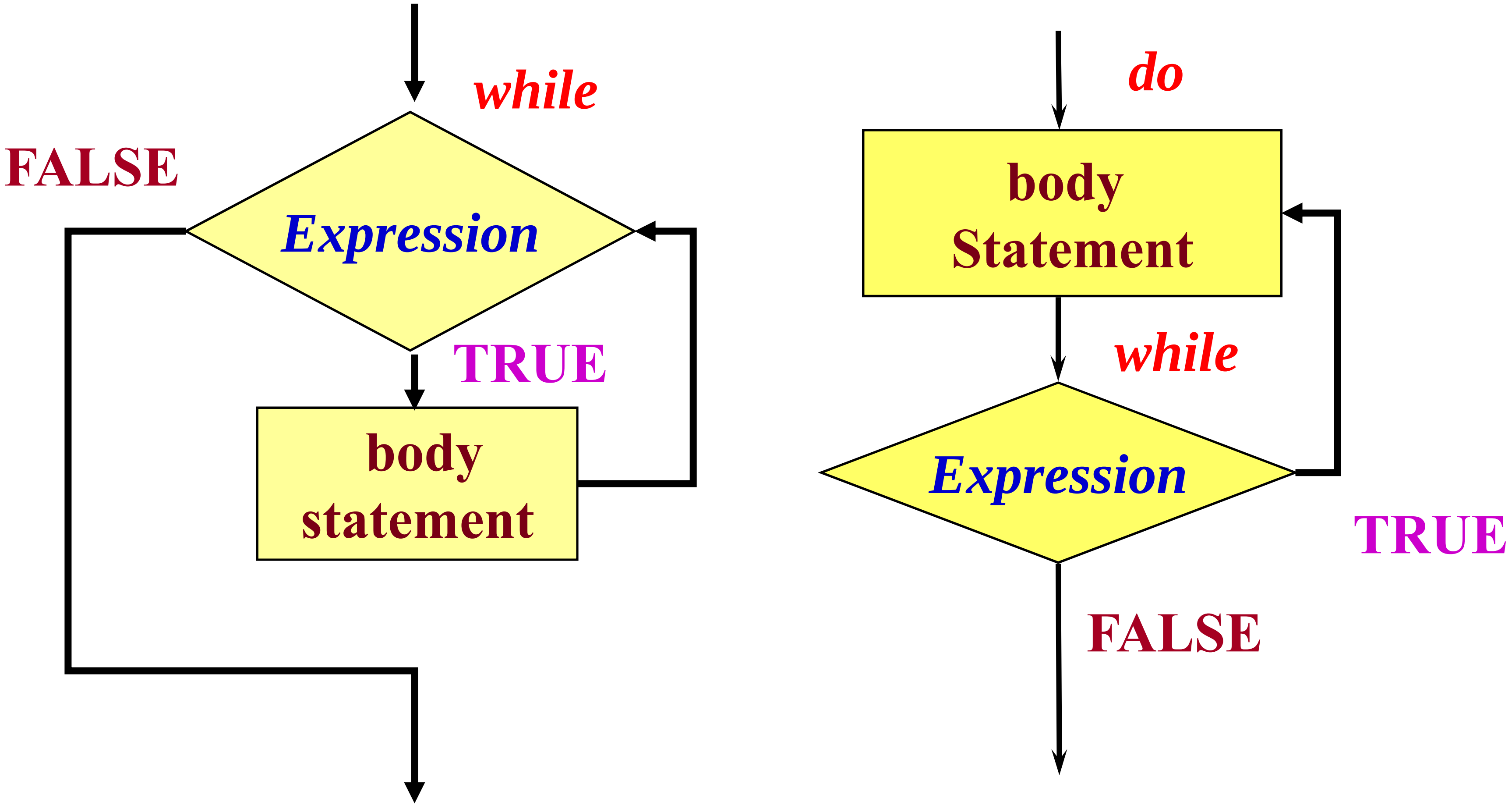


```
   *
  ***
 *****
*******
 *****
  ***
   *
```



```
 1   2   3   4   5   6   7   8   9
     4   6   8  10  12  14  16  18
         9  12  15  18  21  24  27
            16  20  24  28  32  36
                25  30  35  40  45
                    36  42  48  54
                        49  56  63
                            64  72
                                81
```

4.9 *do-while* *Loop* (Page 156)



4.9 *do-while* Loop (Page 156)

- General form

```
 do{
     statement1;
     statement2;
     . . .
  }while (expression);
```

- Statements between the braces are executed at least once

```
while (expression)
{
    statement1;
    statement2;
    . . .
 };
```

- Statements between the braces may be executed

Generate Factorial n!=1*2*3*…*n

```
#include "stdafx.h"
int _tmain(int argc, _TCHAR* argv[])
{
int     i,nFactorial,n;

  n=5;
  nFactorial=1;
  i=1;
  do{
     nFactorial*=i;
     i++;
  }while(i<=n);

  printf("\n%d!=%d\n",n,nFactorial);
  return 0;
}
```

Home Work

- $\sum n! = 1! + 2! + 3! + 4! + 5!$
- Page 165-4 Use a do-while loop and the following....

4.10 *for* *Loop* (Page 156)

- General form

  for **(expression1 ; expression2 ; expression3)**

  statement;

  - **expression1** - **initialization** expression that initializes the ***for*** loop control variable.
  - **expression2** - loop repetition **condition**. Test expression. If False, loop is terminated.
  - **expression3** - **increment** expression that increases or decreases the control variable.

- ***for*** **(i=1 ; i<=3 ; i++)**

```
{
  printf (“i=%2d\n",i);
   i++;
}
```

$$nSum = \sum_{i=1}^{100} i = 1 + 2 + 3 + \ldots + 100$$

```
int     i,nSum=0;


 i=1;
 while(i<101)
 {
    nSum=nSum+i;
    i=i+1;
 }
```

```
int     i,nSum=0;

    for(i=1;i<101;i++)
       nSum=nSum+i;
```

```
nSum?=0;
for(i?=1;i<1?01;i+?+)
   nSum?=nSum+i;
```

```
nSum=0;i=1;
for(    ;i<101;i++)
   nSum=nSum+i;
```

```
nSum=0; i=1;
for(    ;i<101;    )
{
    nSum=nSum+i;
    i++;
}
```

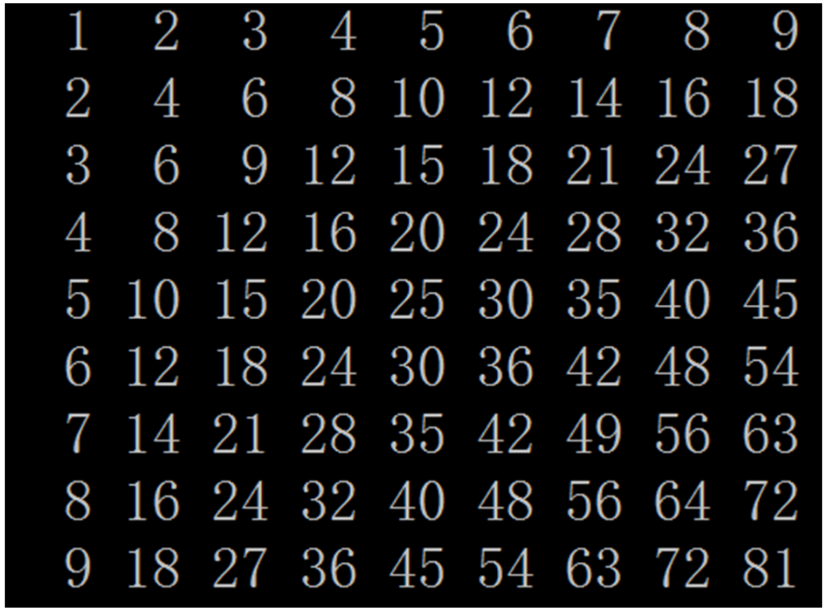
```
nSum=0; i=1;
for(    ;        ;    )
{
    nSum=nSum+i;
    i++;
    if(i>100)
        break;
}
```

```
for(nSum=0,i=1;i<101;i++)
    nSum=nSum+i;
```

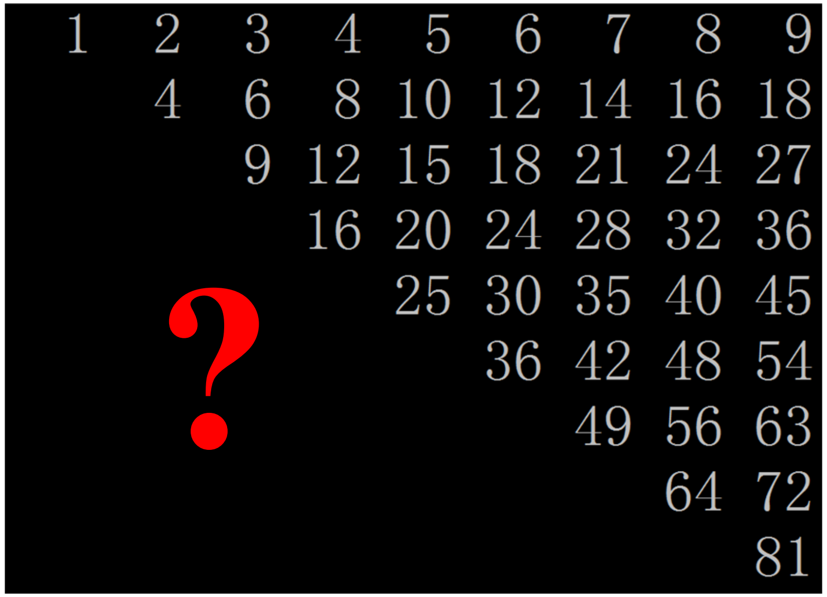
```
for(nSum=0,i=1;i<101;nSum=nSum+i,i++)
   ;
```

Nested *for* Loop

```
int     i,j;
  for(i=1;i<10;i++)
  {
    for(j=1;j<10;j++)
       printf("%3d",i*j);
    printf("\n");
  }
```



```
  1  2  3  4  5  6  7  8  9
  2  4  6  8 10 12 14 16 18
  3  6  9 12 15 18 21 24 27
  4  8 12 16 20 24 28 32 36
  5 10 15 20 25 30 35 40 45
  6 12 18 24 30 36 42 48 54
  7 14 21 28 35 42 49 56 63
  8 16 24 32 40 48 56 64 72
  9 18 27 36 45 54 63 72 81
```



```
  1  2  3  4  5  6  7  8  9
     4  6  8 10 12 14 16 18
        9 12 15 18 21 24 27
          16 20 24 28 32 36
             25 30 35 40 45
                36 42 48 54
                   49 56 63
                      64 72
                         81
```

break Statement

- A *break* statement forces an immediate break, or exit, from *switch*, *while*, *for*, and *do-while* statements.
- If *break* is in a loop that is nested inside another loop, control exits the inner loop but not the outer.

```
int   i,nMax,nNum;//Find Primes

  nMax=40;
  for(nNum=2;nNum<nMax;nNum++)
  {
     i=2;
     while(i<nNum)
     {
        if(nNum%i==0)
           break;
        i++;
     }
     if(i==nNum)
        printf("%d\n",nNum);
  }
```

continue Statement

- When a *continue* is encountered in a loop, the next iteration of the loop begins immediately.
- For *while* loops this means that execution is automatically transferred to the top of the loop and reevaluation of the tested expression is initiated.

```
int  i,nMax;    //x%3!=0
  i=1;nMax=20;
  while(i<nMax);
  {
     i++;
     if(i%3==0)
        continue;
     printf("%d_",i);
  }
```

Home Work

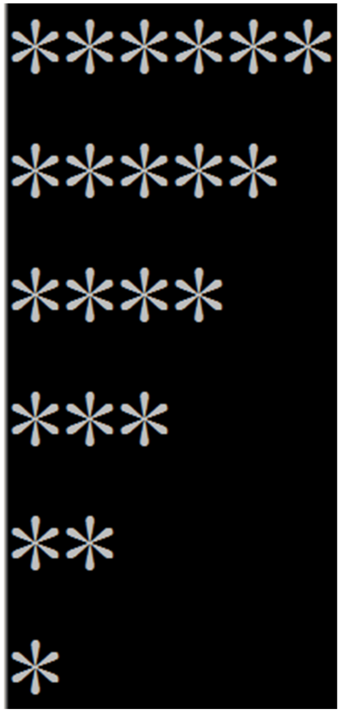
- Page 138-2 Write a program to input these ten incomes from a file
- Page 141-1 Given x = 200 andy= -400， determine whether…
- Page 141-3 Suppose the yearly demand…
- Page 155-3 Use a switch statement to write a program...
- Page 159-3 Use a while loop to show that...
- Page 159-4 Use if and other C statements to determine...
- Page 162-2 Use a while loop to calculate the factorial....
- Page 165-4 Use a do-while loop and the following.... ∑n!=1!+2!+3!+4!+5!

Home Work

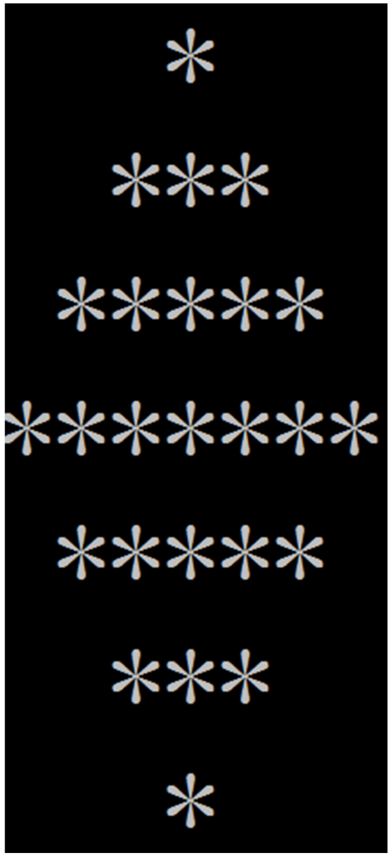
- Page175-3 Hand calculate the final value of…
- Page175-4 Write a program to calculate the mean…
- Page185 Write a program to calculate the mean…

  But print the second minimum of x^2+y^2 under the constraint :$x*y^2=54$.
- Page197-4.6 Write a program that is capable…
- Get all Prime numbers(素数)between 100 and 200

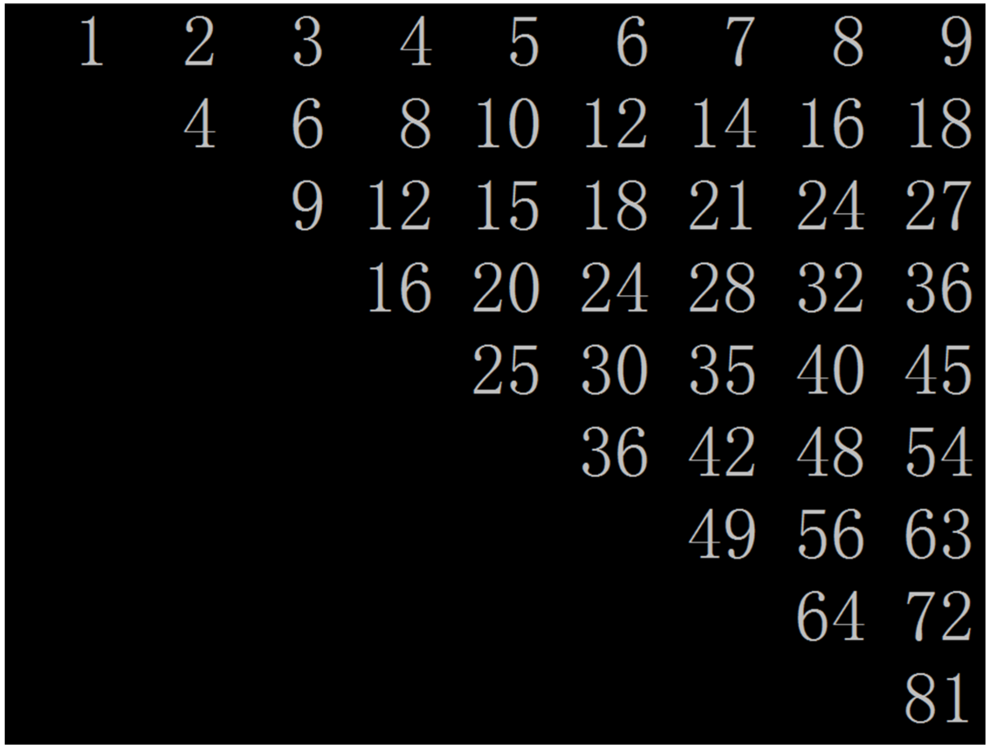
Home Work



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



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



```
  1   2   3   4   5   6   7   8   9
      4   6   8  10  12  14  16  18
          9  12  15  18  21  24  27
             16  20  24  28  32  36
                 25  30  35  40  45
                     36  42  48  54
                         49  56  63
                             64  72
                                 81
```