Introduction to C Programming

Lecture 3: decision & looping

Wenjin Wang
wangwangj3@sustech.edu.cn

9-23-2022

Course syllabus

Nr.	Lecture	Date
1	Introduction	2022.9.9
2	Basics	2022.9.16
3	Decision and looping	2022.9.23
4	Array & string	2022.9.30
5	Functions	2022.10.9 (补)
6	Pointer	2022.10.14
7	Self-defined types	2022.10.21
8	Memory control & file I/O	2022.10.28

Nr.	Lecture	Date
9	Head files & pre-processors	2022.11.4
10	Review of lectures	2022.11.11
11	Soul of programming: Algorithms I	2022.11.25
12	Soul of programming: Algorithms II	2022.12.2
13	R&D project	2022.12.9
14	R&D project	2022.12.16
15	R&D project	2022.12.23
16	Summary	2023.12.30

Recap last lecture

- **Bit** is the atomic unit (0,1) for data storage, **byte (8 bits)** is the smallest unit for information storage
- Different **data types** (int, float, double, char) can be used to declare variables (placeholders for data)
- Five basic **operations** provided by C can manipulate variables: arithmetic, relational, logical, assignment, Misc
- You can interact with machine using **I/O functions** (scanf, printf)
- You can use C to make basic calculations with I/O!

Recap last lecture

```
#include<stdio.h>

main()
{
  int a, b;
  printf(“Enter two numbers:");
  scanf("%d, %d", &a, &b);

  int c = a + b;
  printf("a + b = %d\n", c);
}
```

Declare variable

Initialize variable

Calculate

Recap last lecture

1

```
main()
{
   //do nothing!
}
```

2

```
main()
{
   int a, b;
}
```

3

```
#include<stdio.h>
main()
{
    int a, b;
    printf(“Enter two numbers:");
    scanf("%d, %d", &a, &b);
}
```

4

```
#include<stdio.h>
main()
{
    int a, b;
    printf(“Enter two numbers:");
    scanf("%d, %d", &a, &b);

    int c = a + b;
}
```

5

```
#include<stdio.h>
main()
{
    int a, b;
    printf(“Enter two numbers:");
    scanf("%d, %d", &a, &b);

    int c = a + b;
    printf("a + b = %d\n", c);
}
```

Recap last lecture

```
scanf(formatted text, variables);
```

```
int a;
float b;
char c;
scanf(“%d, %f, %c\n”, &a, &b, &c);
```

```
printf(formatted text, variables);
```

```
printf(“%d, %f, %c\n”, a, b, c);
```

Recap last lecture

How to use symbol ?

Get maximum

```
int a = 3, b = 8, c = 1;

int max = a;
max = max > b ? max : b;
max = max > c ? max : c;

printf(“max = %d”, max);
```

Get minimum

```
int a = 2, b = 6, c = 3;

int min = a;
min = min < b ? min : b;
min = min < c ? min : c;

printf(“min = %d”, min);
```

Objective of this lecture

You can use C to control the workflow!

Content

1. Decision-making (if, switch)
2. Looping (for, while)

Content

1. Decision-making (if, switch)
2. Looping (for, while)

Decision-making in life

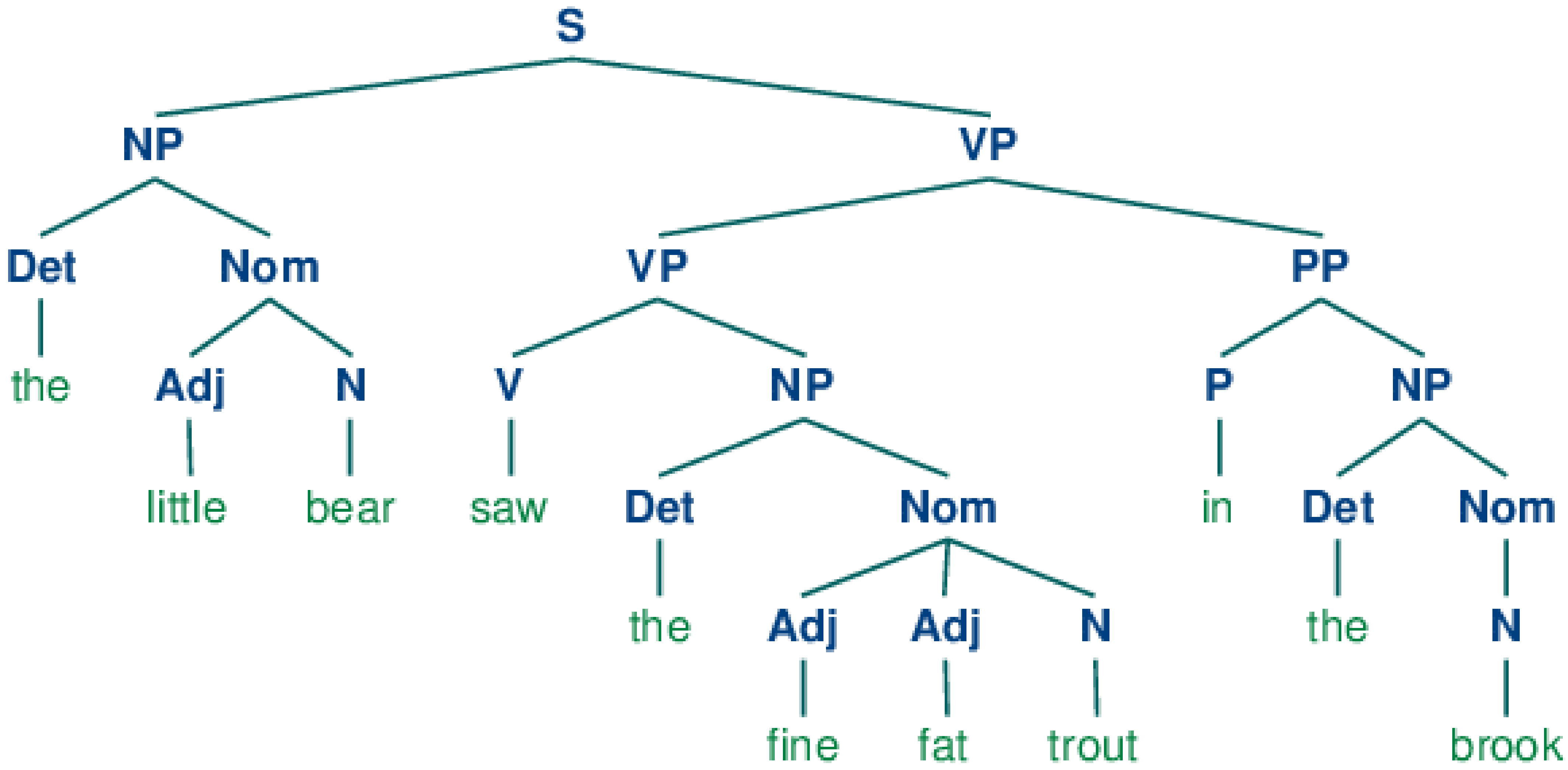
“I love fish, I also love palm of bear. But I cannot get both, so I take the palm of bear!”

“鱼，我所欲也；熊掌，亦我所欲也。二者不可兼得，舍鱼而取熊掌也。”

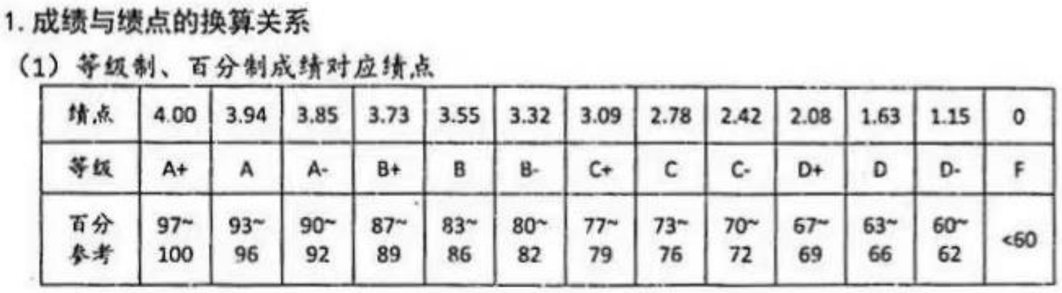
Decision-making in life

Decision-making in life

Decision-making in life



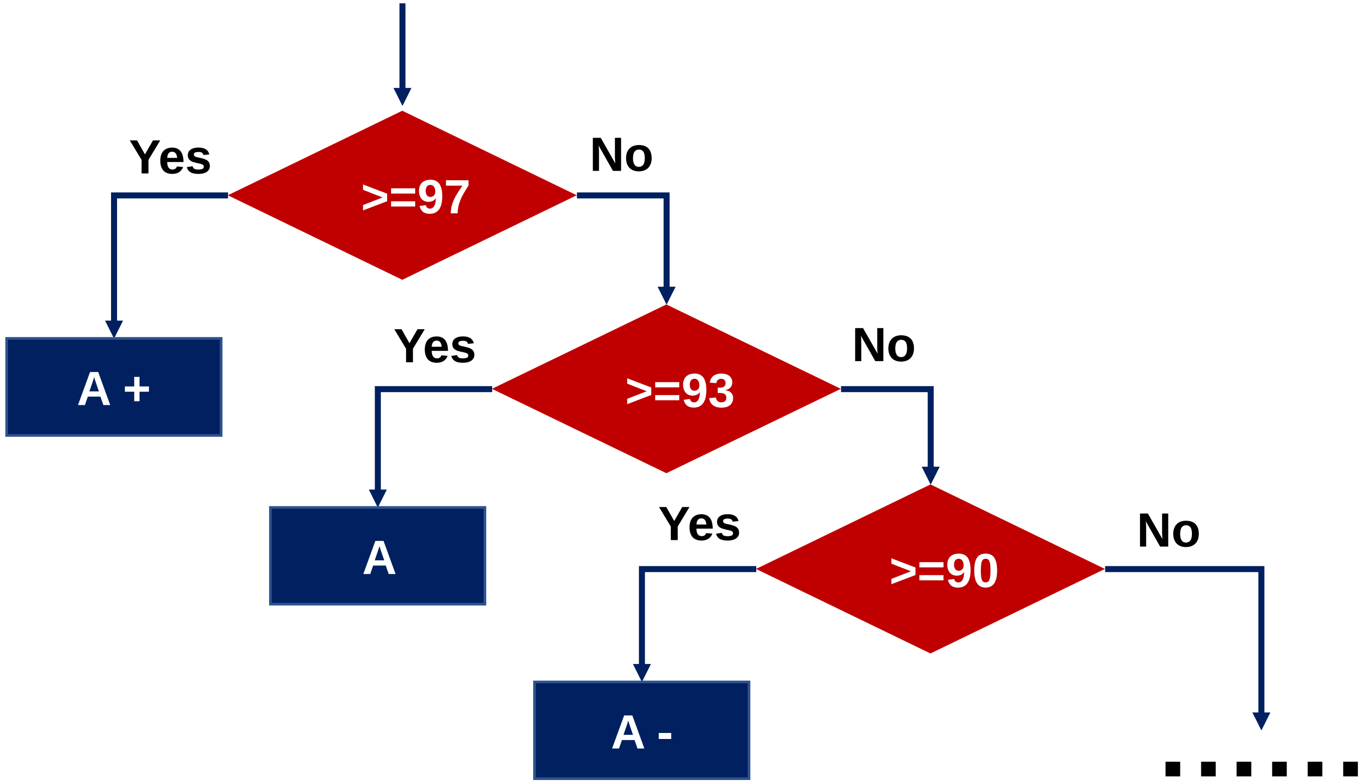
Decision-making in life



1. 成绩与绩点的换算关系

（1）等级制、百分制成绩对应绩点

绩点	4.00	3.94	3.85	3.73	3.55	3.32	3.09	2.78	2.42	2.08	1.63	1.15	0
等级	A+	A	A-	B+	B	B-	C+	C	C-	D+	D	D-	F
百分参考	97~100	93~96	90~92	87~89	83~86	80~82	77~79	73~76	70~72	67~69	63~66	60~62	<60



Yes

>=97

No

A +

Yes

>=93

No

A

Yes

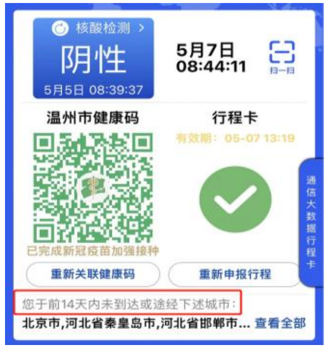
>=90

No

A -

……

Decision-making in life


核酸检测 >

阴性

5月5日 08:39:37

5月7日
08:44:11

温州市健康码

已完成新冠疫苗加强接种

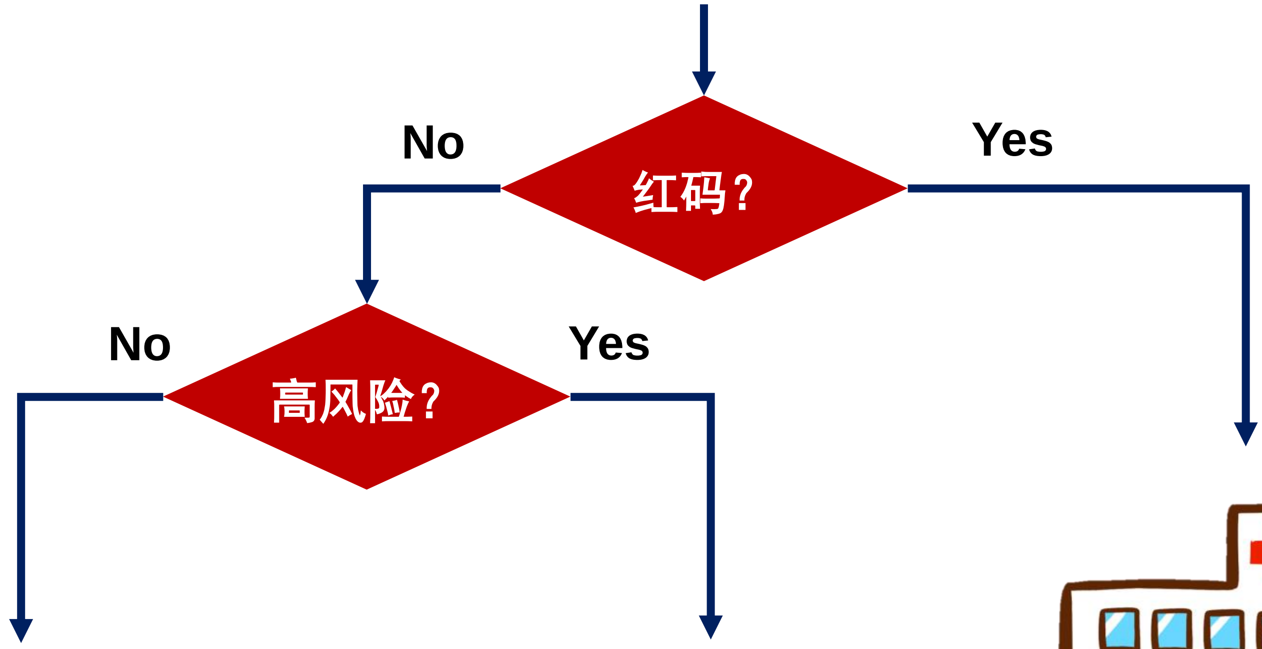
行程卡

有效期：05-07 13:19

通信大数据行程卡

您于前14天内未到达或途经下述城市：

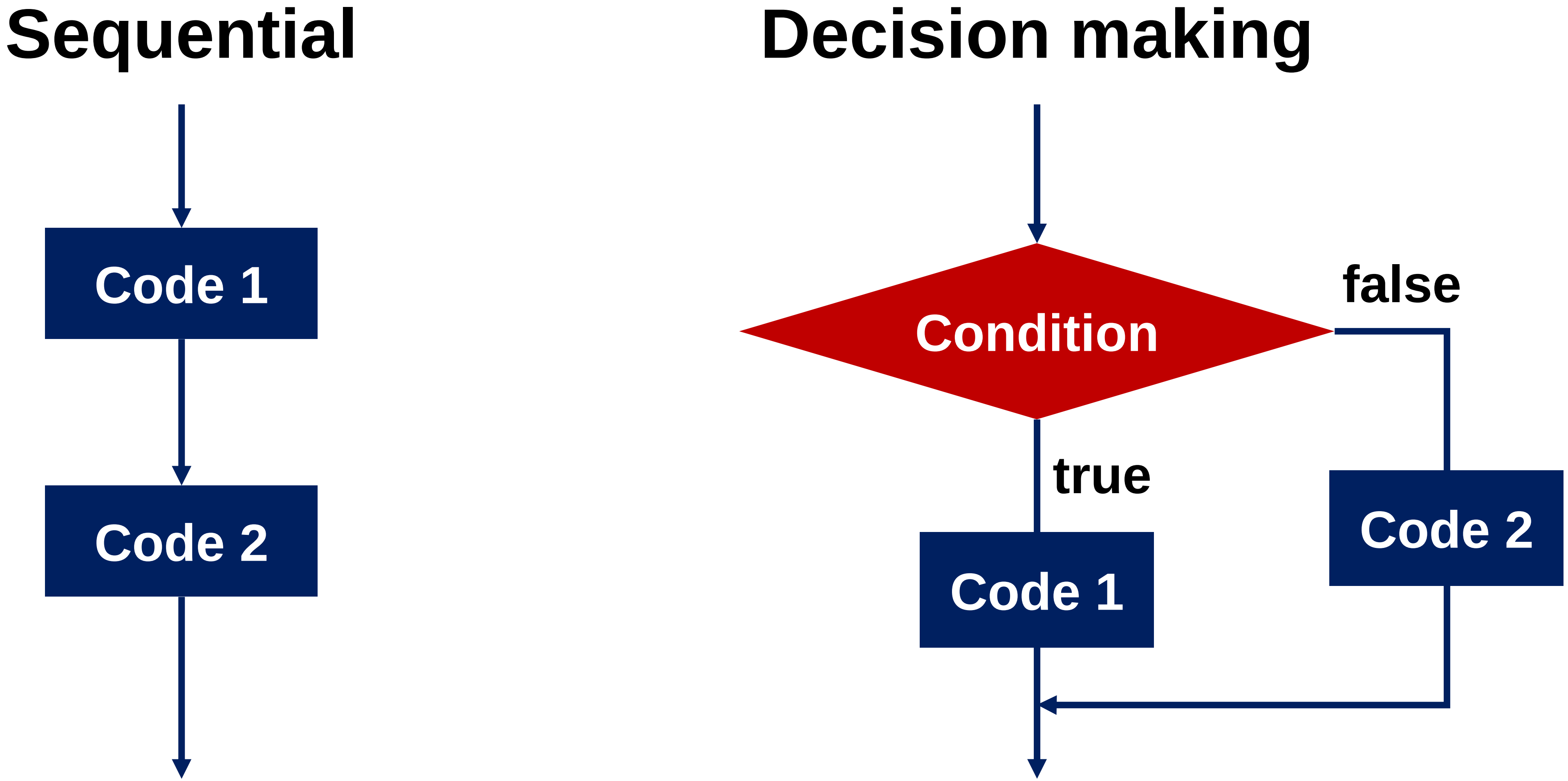
北京市,河北省秦皇岛市,河北省邯郸市...



Free to go

Quarantine at home (3 days!)

Decision-making in program



If statement

If statement has a boolean expression followed by one or more statements.

```
if(boolean_expression)
{ /* code 1 */ }
```

```
if(boolean_expression)
{ /* code 1 */ }
else
{ /* code 2 */ }
```

If statement

if(condition)	**如果**（条件满足）
{option A}	{A选项}
else	**否则**
{option B}	{B选项}

If statement

condition

Ture/Yes/1

(2<5, 3==3, ’a’!=‘b’)

False/No/0

(1>=5, 3<1, ’a’==‘b’)

If and if-else

What does condition mean?

```
int a = 3;

if (a > 10)
{
  // …
}else
{
  // …
}
```

```
float f = 10;

if (a == 10)
{
  // …
}else
{
  // …
}
```

```
char c = ‘A’;

if (a != ‘A’)
{
  // …
}else
{
  // …
}
```

If and if-else

If only

```
int a = 3;

if (a > 10)
{
   printf(“a > 10”);
}

return 0;
```

Block 1

if else

```
int a = 3;

if (a > 10)
{
   printf(“a > 10”);
}else{
   printf(“a < 10”);
}

return 0;
```

Block 1

Block 2

If versus ?

If statement

```
#include <stdio.h>

main ()
{
  int a = 5, b = 10;

  if(a < b)
  {
    printf(“b is larger!");
    printf(“b is %d“, b);
    b++;
    //…
  }
}
```

More space to do things!

? statement

```
#include <stdio.h>

main ()
{
  int a = 5, b = 10;

  int max = a < b ? b : a;
  printf(“max is %d“, max);
}
```

Can only set one variable!

If versus ?

If statement

```
#include <stdio.h>

main ()
{
  int a = 5, b = 10, c = 20;

  if(a < b && a < c && b < c)
  {
    printf(“c is larger!");
    printf(“c is %d“, c);
    b++;
    //…
  }
}
```

Set multiple conditions!

? statement

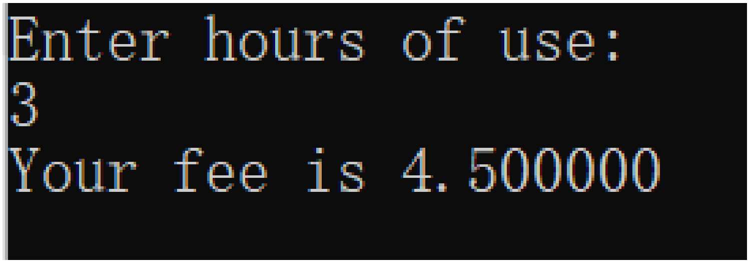
```
#include <stdio.h>

main ()
{
  int a = 5, b = 10;

  int max = a < b ? b : a;
  printf(“max is %d“, max);
}
```

Can only compare two numbers!

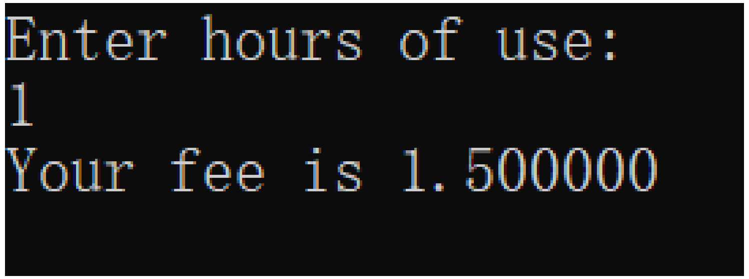
Case study: If statement

Case: calculate the shared bike fee (<1h is 1h)!

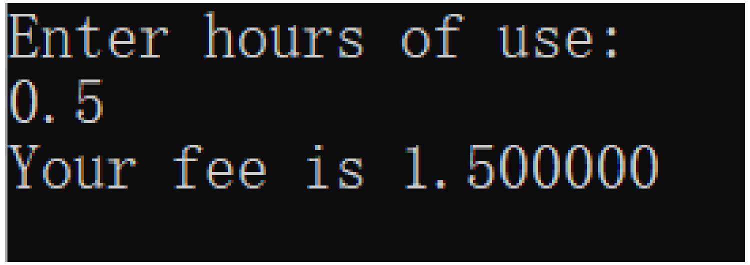
```
#include <stdio.h>
main ()
{
  float hours, fee;
  printf(“Enter hours of use:\n");
  scanf("%f", &hours);
  if(hours < 1)
  {
     hours = 1;
  }
  fee = 1.5 * hours;
  printf("Your fee is %f", fee);
}
```



```
Enter hours of use:
3
Your fee is 4.500000
```



```
Enter hours of use:
1
Your fee is 1.500000
```

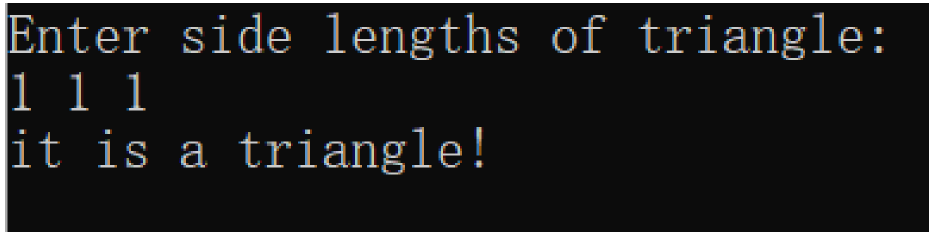


```
Enter hours of use:
0.5
Your fee is 1.500000
```

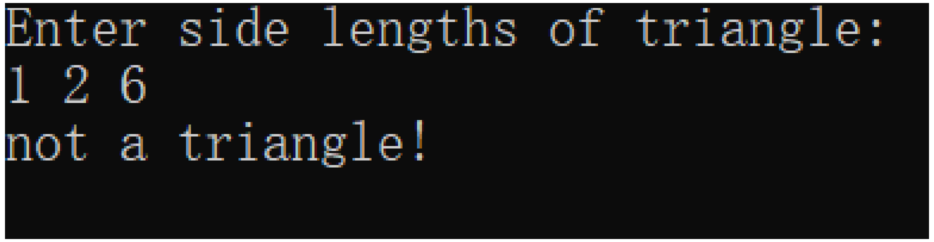
Case study: if-else statement

Case: check if three sides can form a triangle

```c
#include <stdio.h>
main()
{
    float a,b,c;
    printf(“Enter side lengths of triangle:\n");
    scanf("%f %f %f", &a, &b, &c);

    if(a+b>c && a+c>b && b+c>a)
    {
        printf("it is a triangle!");
    }else
    {
        printf("not a triangle!");
    }
}
```



```
Enter side lengths of triangle:
1 1 1
it is a triangle!
```



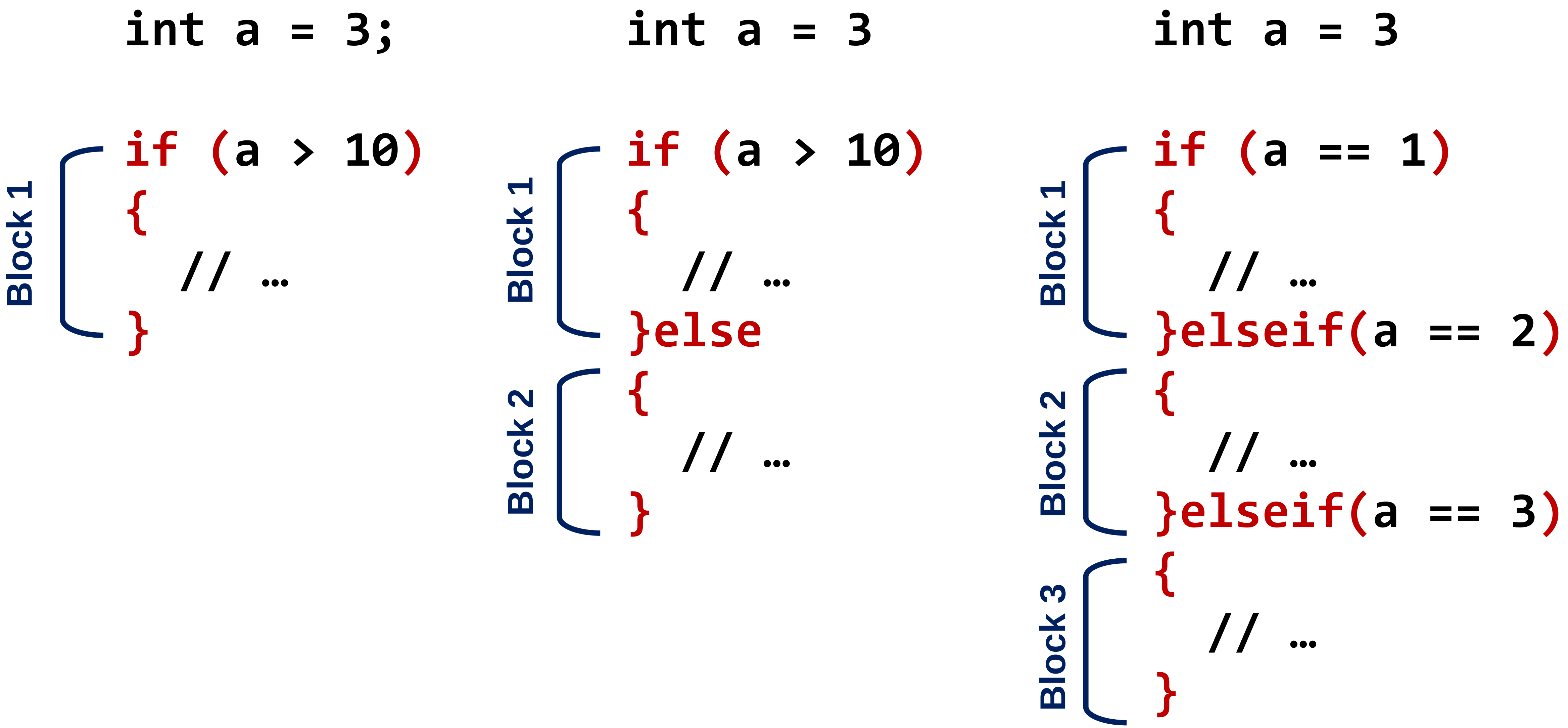
```
Enter side lengths of triangle:
1 2 6
not a triangle!
```

If-elseif

If-elseif has more boolean expression followed by more statements.

```
if( condition 1 )
{ /* code 1 */ }
elseif( condition 2 )
{ /* code 2 */ }
elseif( condition 3 )
{ /* code 3 */ }
elseif( condition 4 )
{ /* code 4 */ }
…
else
{/* code N */}
```

If-elseif



```
int a = 3;

if (a > 10)
{
  // …
}
```

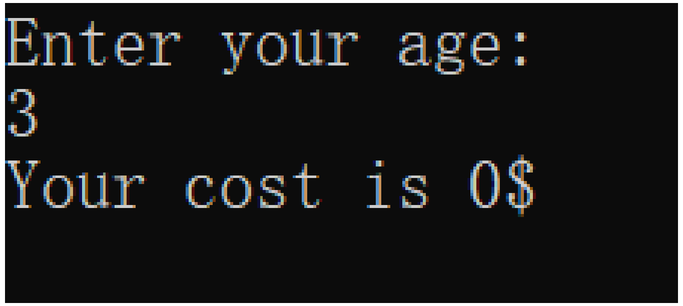
```
int a = 3

if (a > 10)
{
  // …
}else
{
  // …
}
```

```
int a = 3

if (a == 1)
{
  // …
}elseif(a == 2)
{
  // …
}elseif(a == 3)
{
  // …
}
```

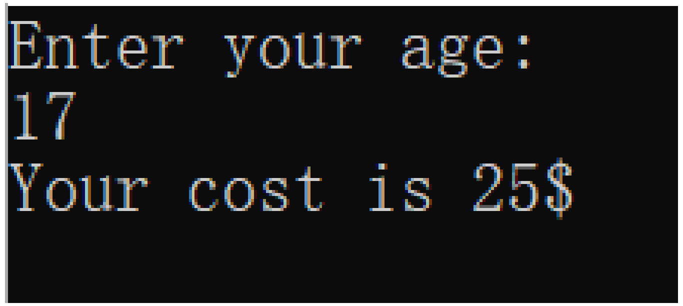
Case study: If-elseif

Case: what is the cost of attendance?

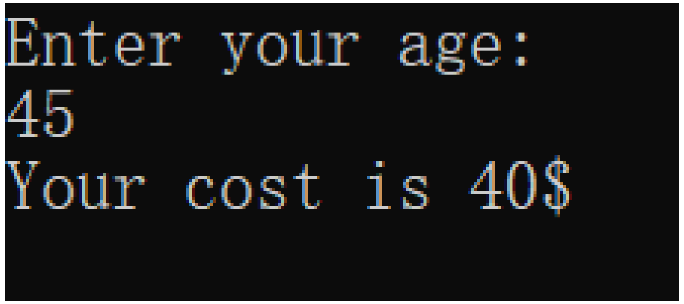
```
#include <stdio.h>
main ()
{
  int a;
  printf(“Enter your age:\n");
  scanf("%d", &a);

  if( a < 10 )
  {
    printf("Your cost is 0$\n" );
  }
  else if( a >= 10 & &a < 20 )
  {
    printf("Your cost is 25$\n" );
  }
  else
  {
    printf("Your cost is 40$\n" );
  }
}
```



```
Enter your age:
3
Your cost is 0$
```



```
Enter your age:
17
Your cost is 25$
```

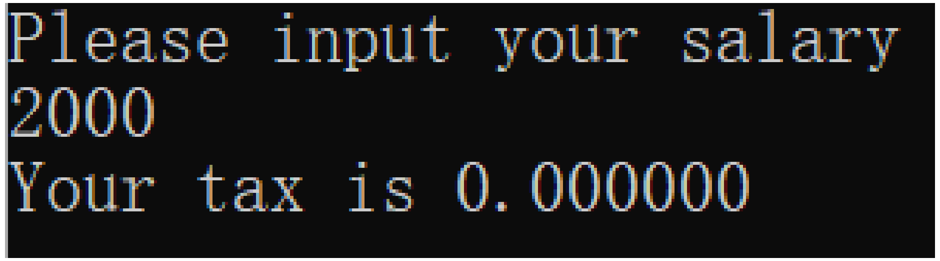


```
Enter your age:
45
Your cost is 40$
```

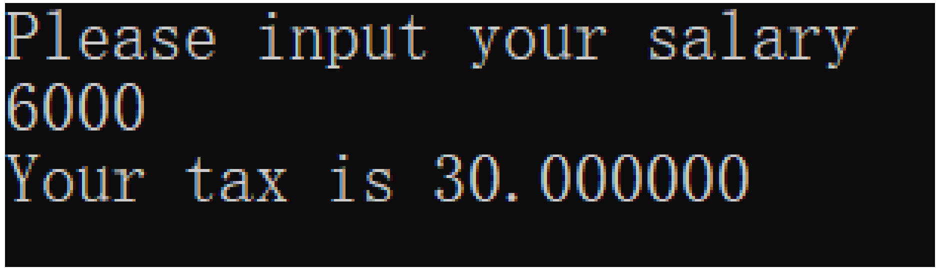
Case study: If-elseif

Case: calculate the tax based on salary

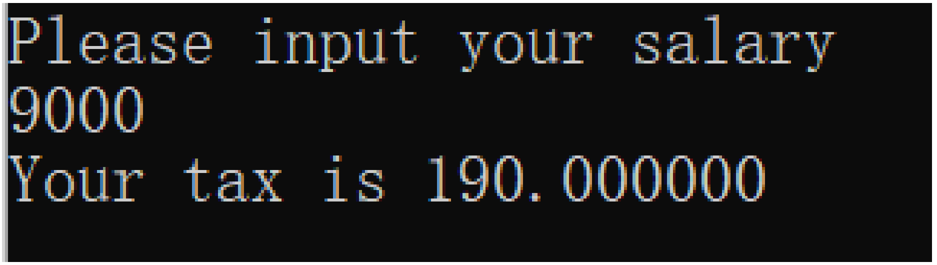
```
#include<stdio.h>
main() {
    double salary, tax;
    printf("Please input your salary\n");
    scanf("%lf", &salary);

    if(salary <= 5000) {
        tax = 0;
    }
    else if(salary <= 8000) {
        tax = (salary - 5000) * 0.03;
    }
    else{
        tax = 90 + (salary - 8000) * 0.1;
    }
    printf("Your tax is %lf\n", tax);
}
```



```
Please input your salary
2000
Your tax is 0.000000
```



```
Please input your salary
6000
Your tax is 30.000000
```



```
Please input your salary
9000
Your tax is 190.000000
```

Nested-if

Nested if-else statement means if can be used inside another if.

```
if( condition 1 )
{
     /* code 1 */
     if( condition 2)
     {
           /* code 2 */
     }
}
```

Nested-if

Parallel if

```
      int a = 3;

      if (a > 10)
      {
1.      // …
      }else
      {
2.      // …
      }
```

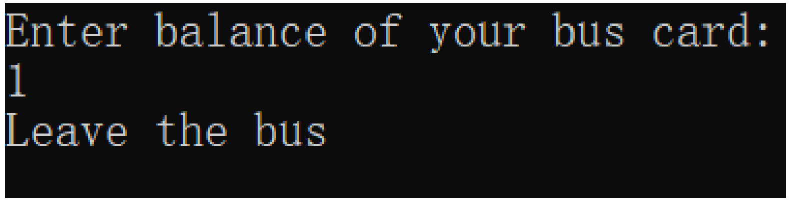
Nested if

```
      int a = 3, b = 10;

      if (a > 10)
      {
              if (b < 5)
              {
1.      1.1       // …
              }
      }else{
2.            // …
      }
```

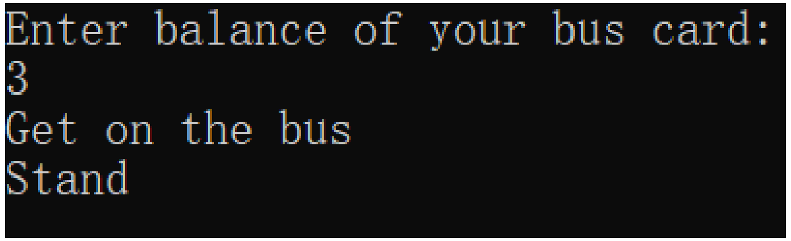
Case study: Nested-if

Case: check the balance of bus card!!!

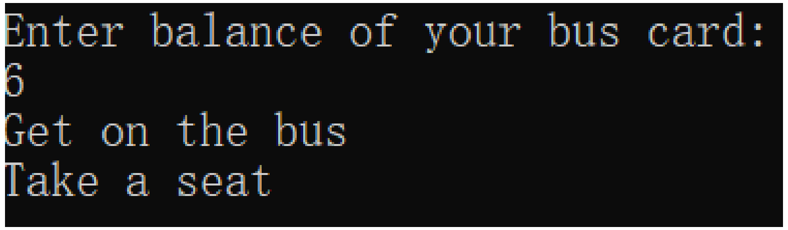
```
#include <stdio.h>
main ()
{
    int a;
    printf(“Enter balance of your bus card:\n");
    scanf("%d", &a);
    if( a >= 2 )
    {
        printf(“Get on the bus\n");
        if( a >=5 )
        {
            printf(“Take a seat\n" );
        }
        else{
            printf(“Stand");
        }
    }
    else{
        printf(“Leave the bus\n");
    }
}
```



```
Enter balance of your bus card:
1
Leave the bus
```



```
Enter balance of your bus card:
3
Get on the bus
Stand
```

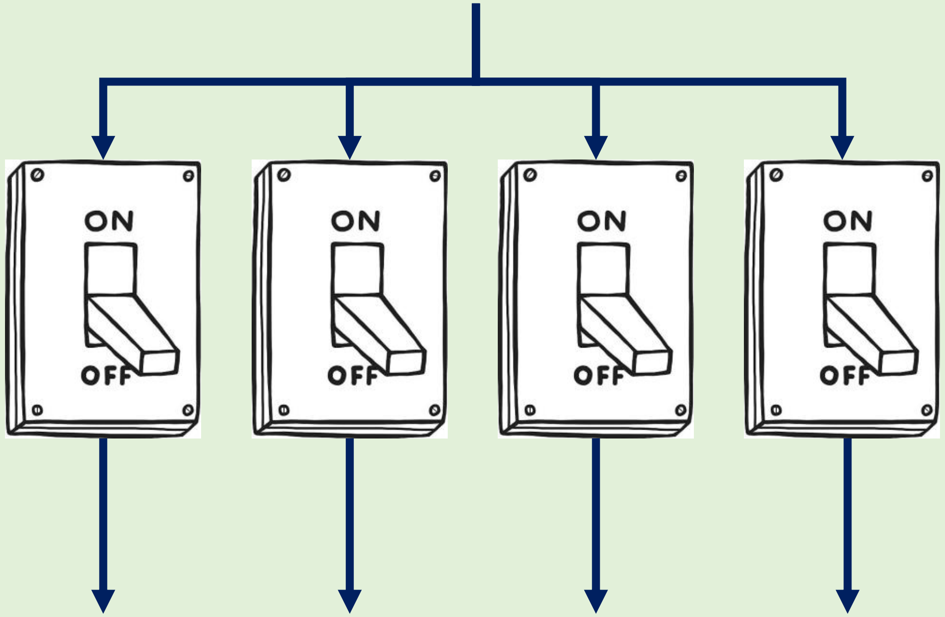


```
Enter balance of your bus card:
6
Get on the bus
Take a seat
```

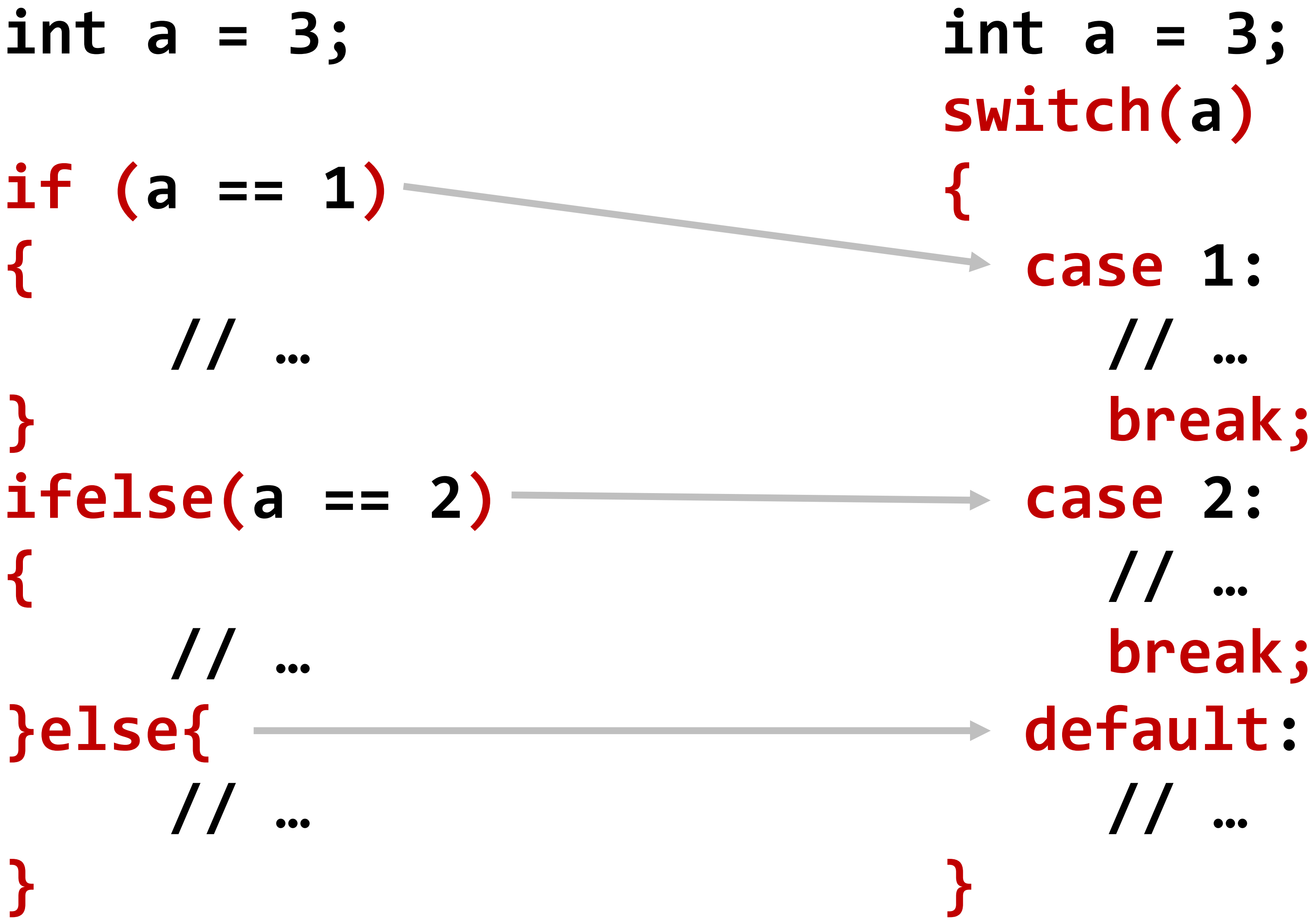
Switch statement

Switch statement allows a variable to be tested for **equality** against a list of values. Case will be switched on if equality meets.

```
switch(variable)
{
   case constant:
      statement;
      break;
   case constant:
      statement;
      break;
   default:
      statement;
}
```



Switch versus if

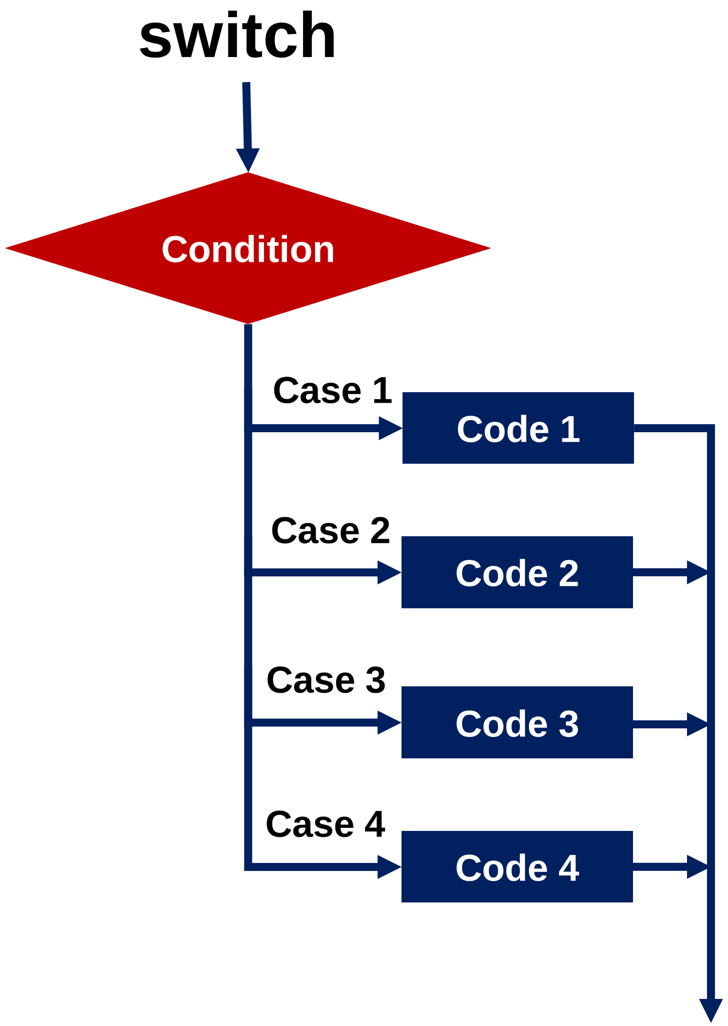


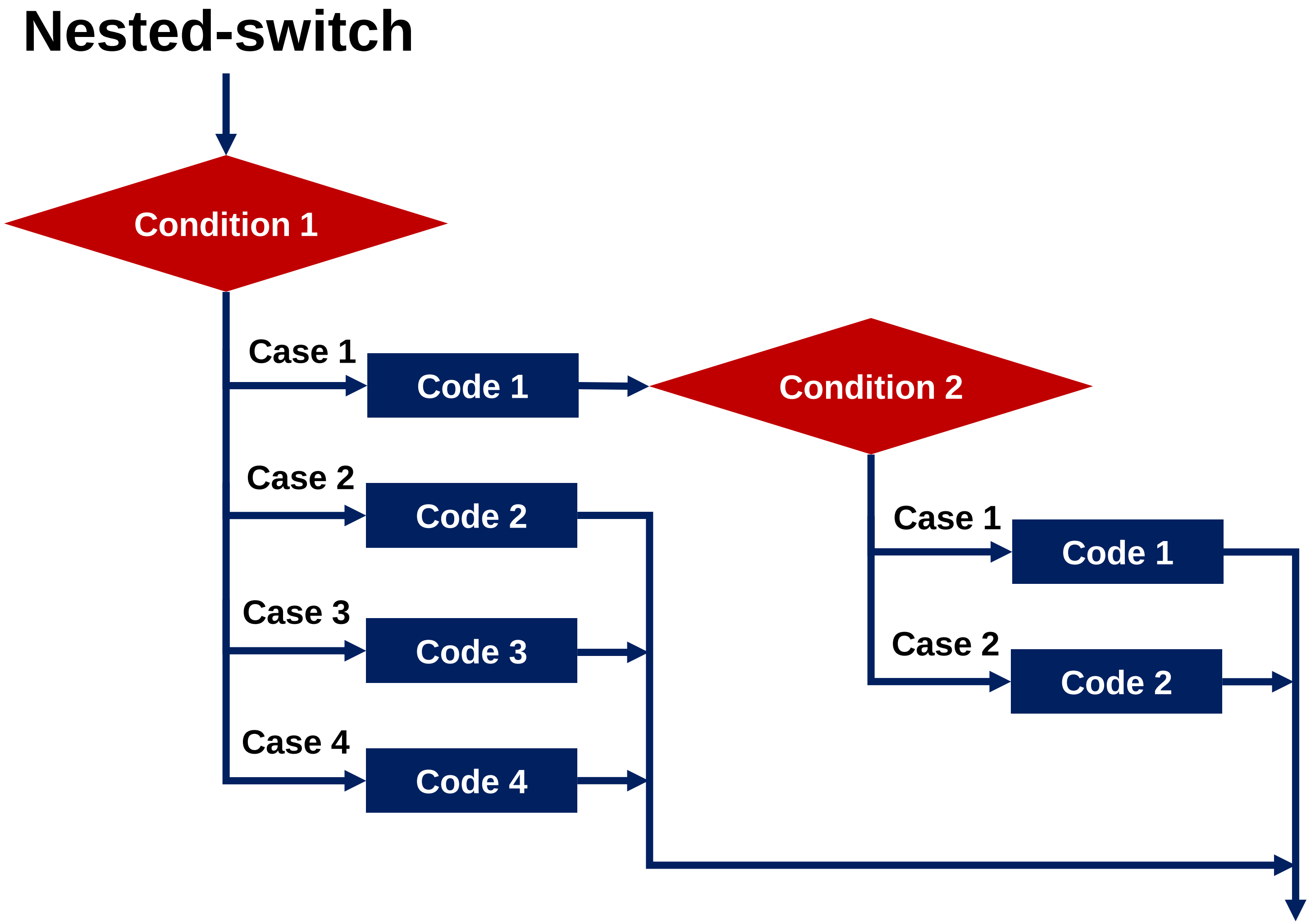
```
int a = 3;

if (a == 1)
{
    // …
}
ifelse(a == 2)
{
    // …
}else{
    // …
}
```

```
int a = 3;
switch(a)
{
  case 1:
    // …
    break;
  case 2:
    // …
    break;
  default:
    // …
}
```

Switch can only express equality!!!

Overview of switch statements

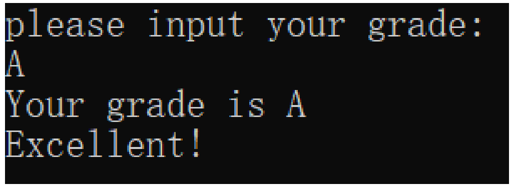




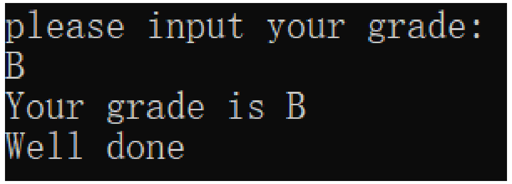
Case study: switch

Case: how to evaluate students based on grades?

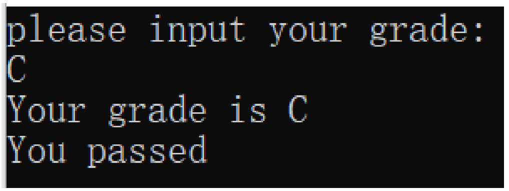
```c
#include <stdio.h>
main ()
{
  char a;
  printf("please input your grade:\n");
  scanf("%c", &a);
  printf("Your grade is %c\n", a );
  switch(a)
  {
     case 'A':
     printf("Excellent!\n" );break;
     case 'B':
     printf("Well done\n" );break;
     case 'C' :
     printf("You passed\n" );break;
     case 'D' :
     printf("Better try again\n" );break;
     default :
     printf("Invalid grade\n" );
  }
}
```



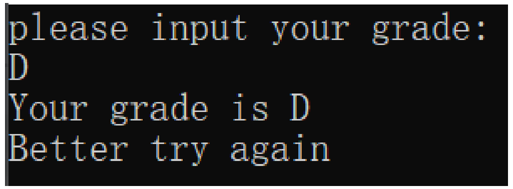
```
please input your grade:
A
Your grade is A
Excellent!
```



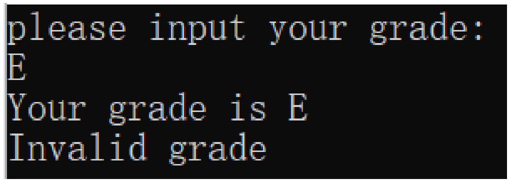
```
please input your grade:
B
Your grade is B
Well done
```



```
please input your grade:
C
Your grade is C
You passed
```



```
please input your grade:
D
Your grade is D
Better try again
```

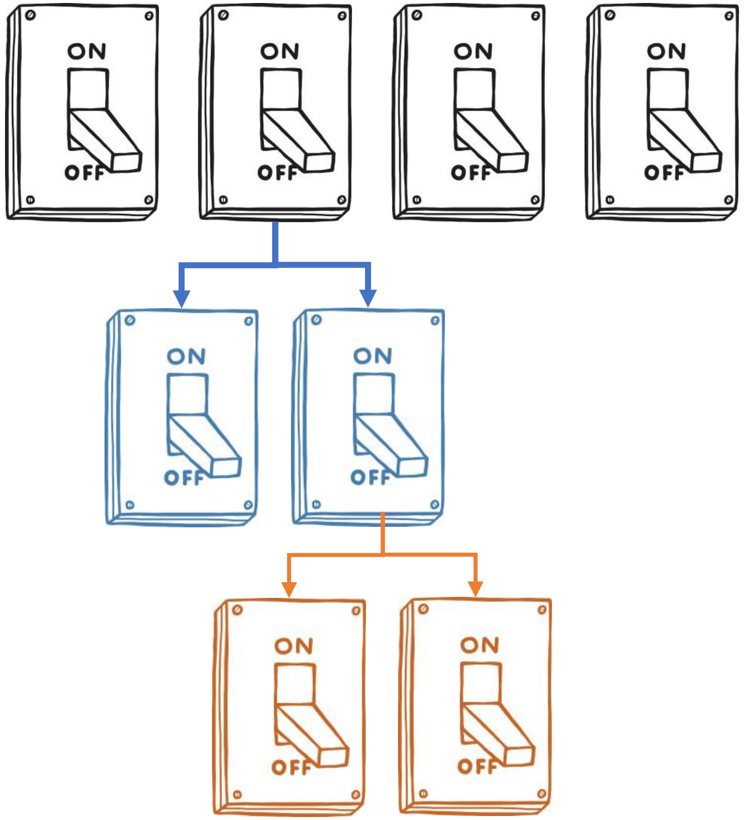


```
please input your grade:
E
Your grade is E
Invalid grade
```

Nested-switch

Switch can be nested. Even if the case constants of the inner and outer switch are the same, no conflict will arise.

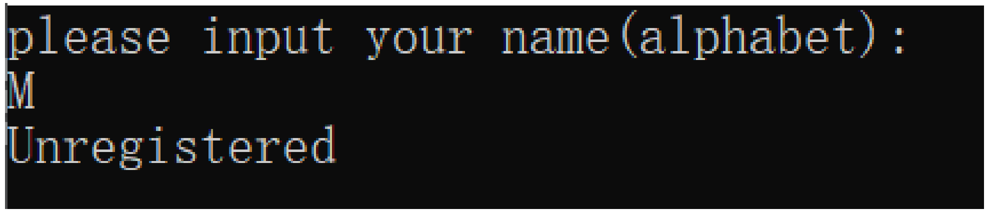
```
switch(ch1) {
case 'A’:
      switch(ch2) {
      case ‘a’:
            statement;
            break;
      case ‘A’:
            statement;
            break
      }
case 'B’:
}
```



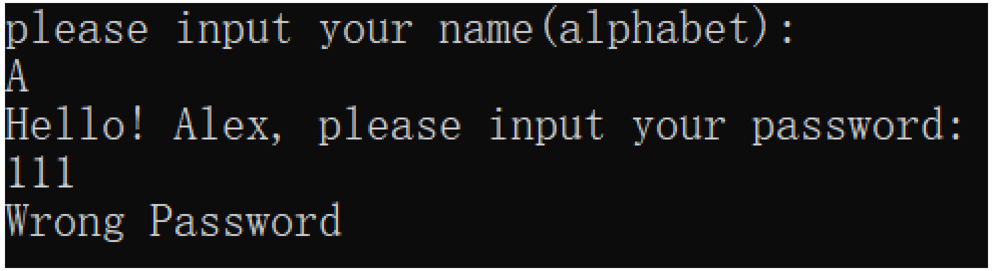
Case study: nested-switch

Case: create a simple login system!

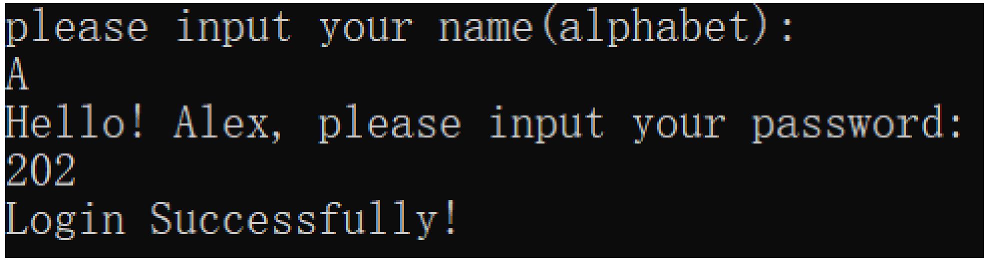
```
#include <stdio.h>
main ()
{
   char a;
   int pw;
   printf("please input your name(alphabet):\n");
   scanf("%c", &a);
   switch(a) {
      case 'A':
         printf("Hello! Alex, please input your password:\n");
         scanf("%d", &pw);
         switch(pw) {
               case 202:
               printf("Login Successfully!");break;
               default:
               printf("Wrong Password\n");
         }break;
      default:
      printf("Unregistered\n" );
   }
}
```



```
please input your name(alphabet):
M
Unregistered
```



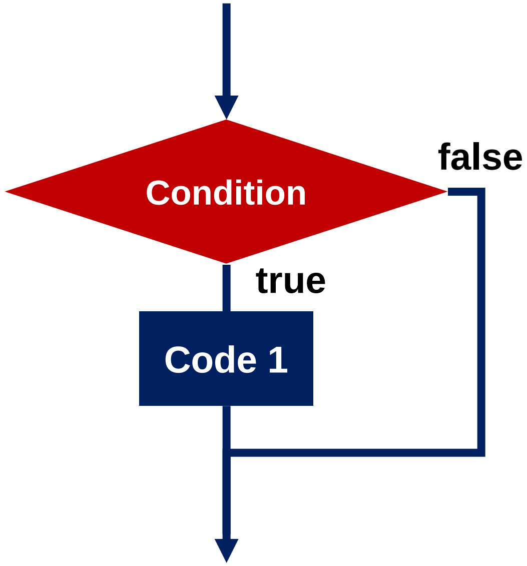
```
please input your name(alphabet):
A
Hello! Alex, please input your password:
111
Wrong Password
```



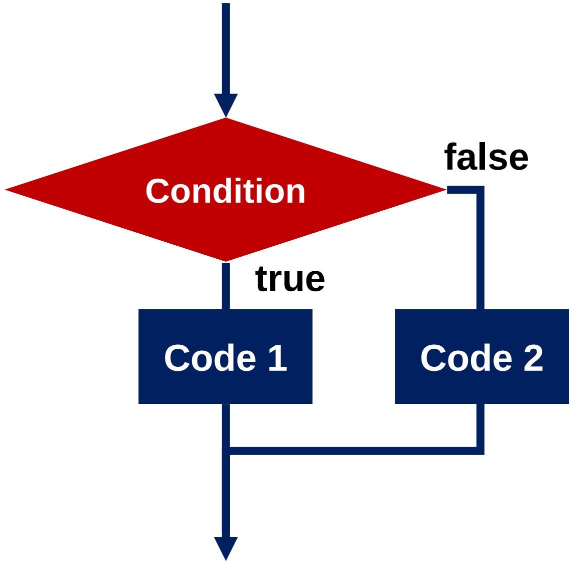
```
please input your name(alphabet):
A
Hello! Alex, please input your password:
202
Login Successfully!
```

Overview of decision-making

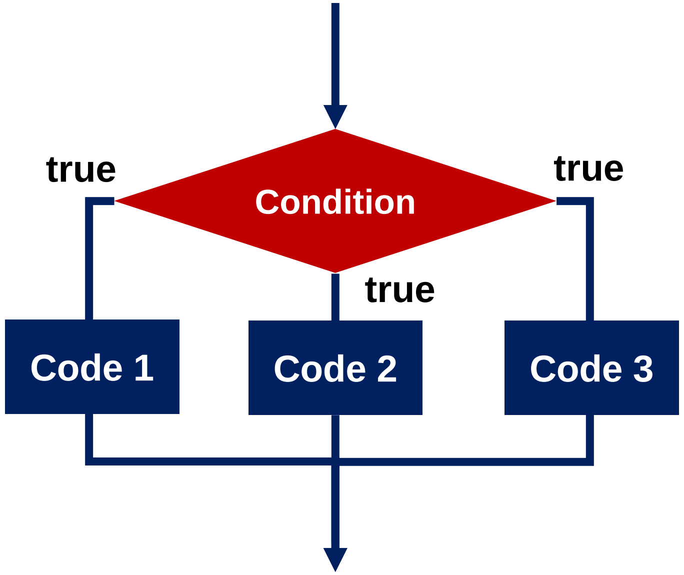
If



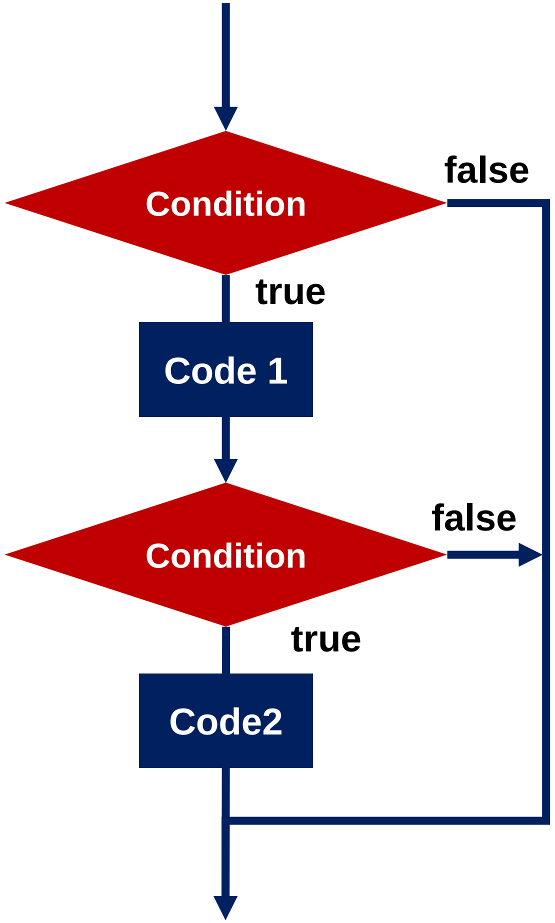
If else



If elseif



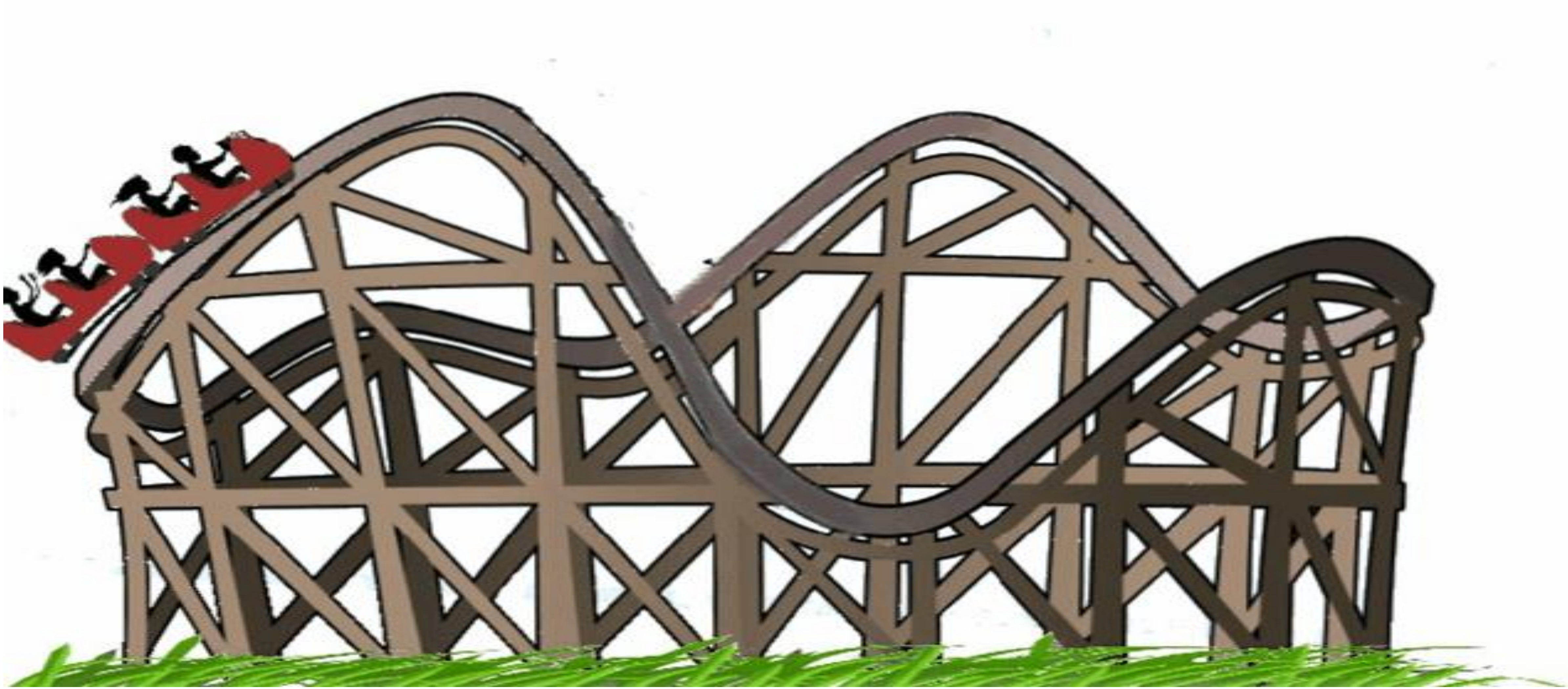
Nested if



Content

1. Decision-making (if, switch)
2. Looping (for, while)

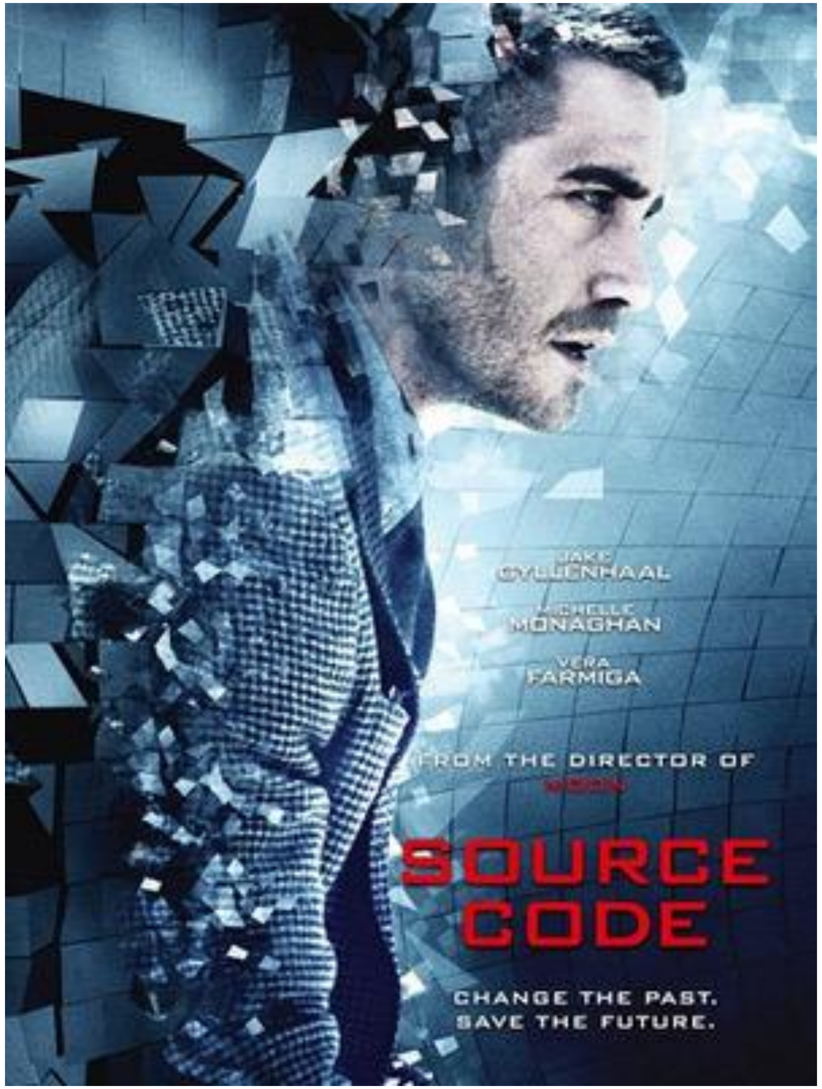
Looping in life



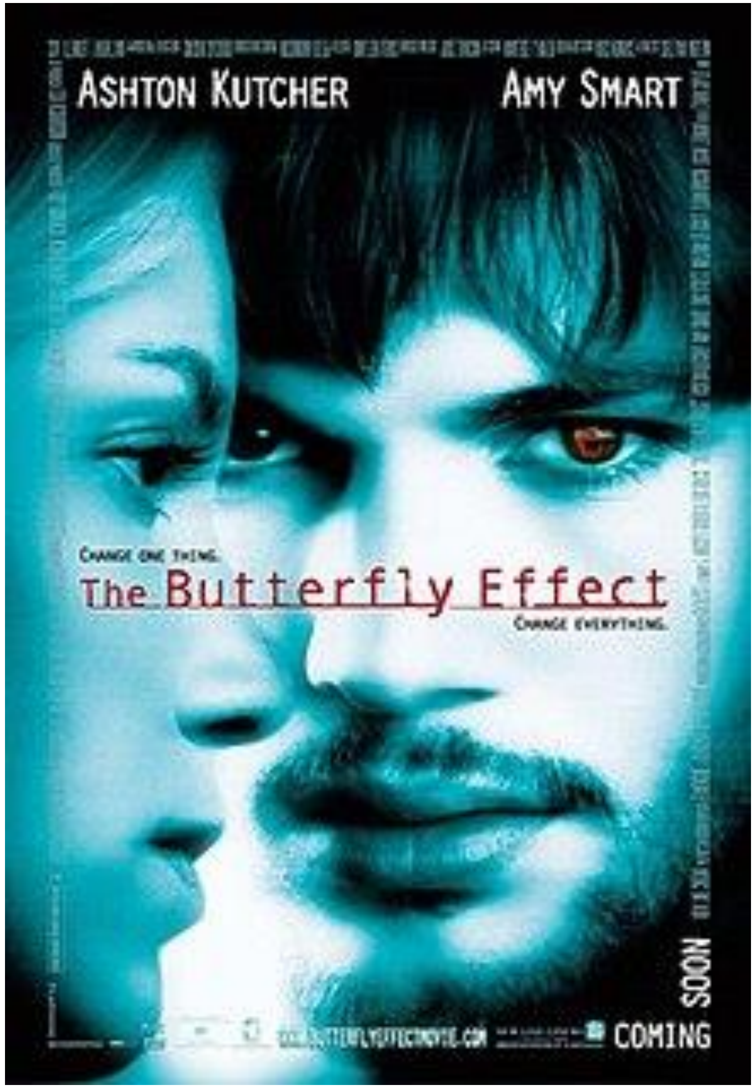
Looping in life

Looping in life

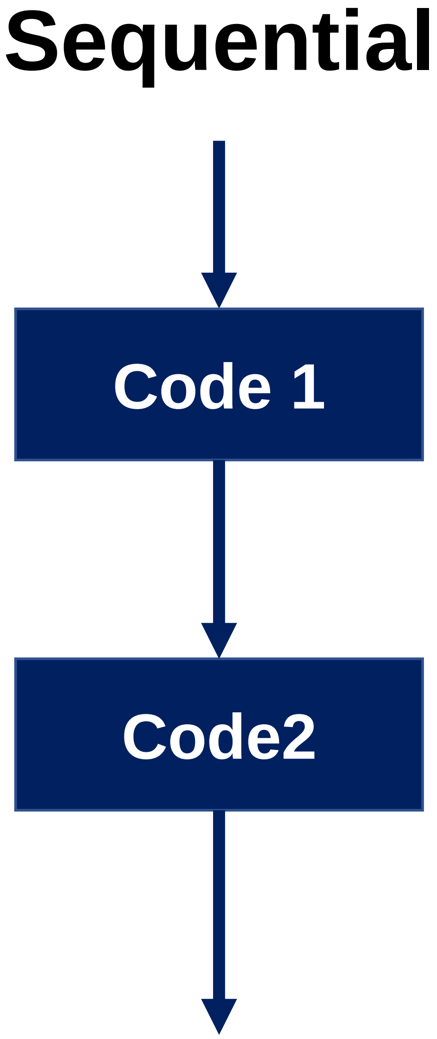
Looping in life

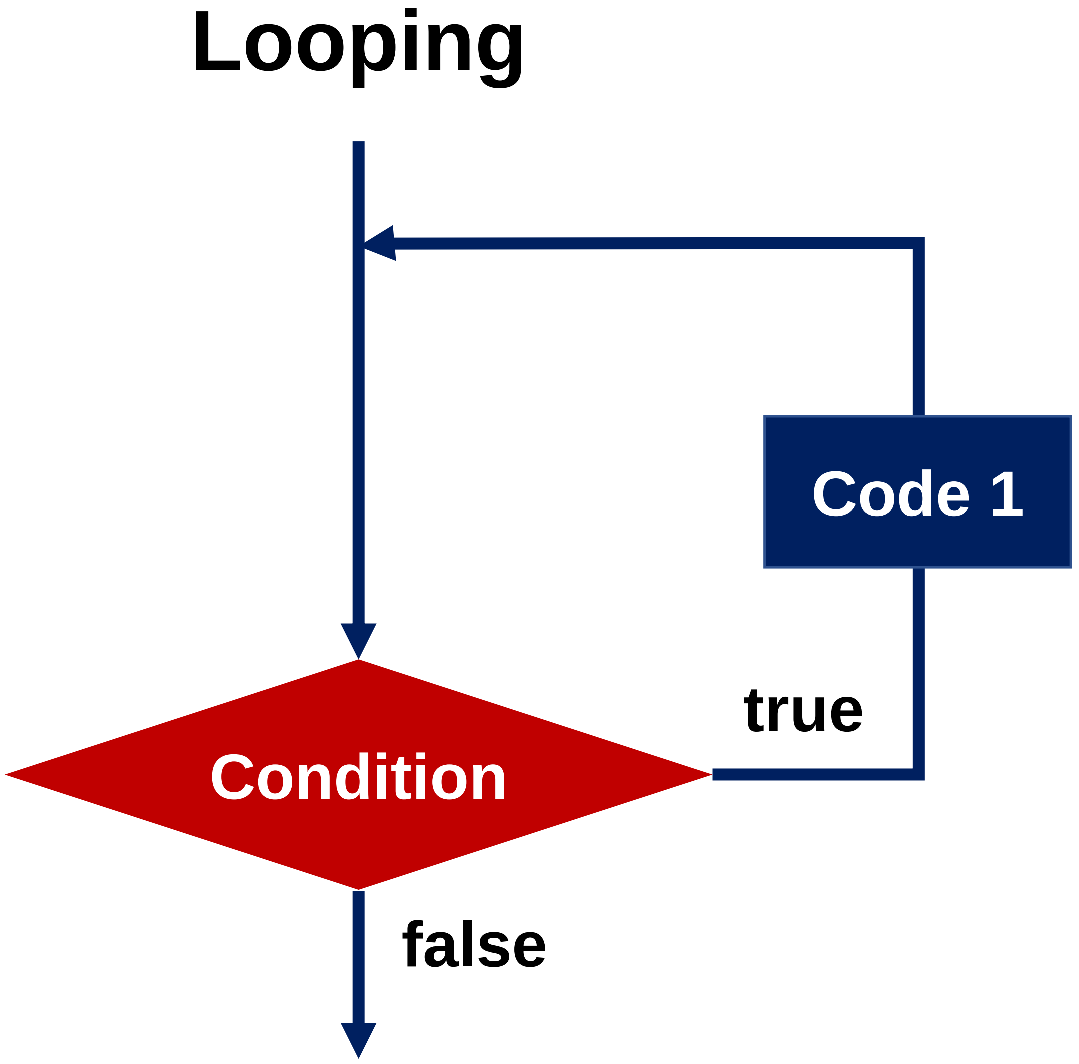


LIVE. DIE. REPEAT.
TOM CRUISE
EMILY BLUNT
EDGE OF TOMORROW
FROM THE DIRECTOR OF THE BOURNE IDENTITY AND MR. & MRS. SMITH



Looping in program





For loop

For loop is a control structure that allows repeating the same operation (but different input values) for a specific number of times.

```
for ( init; condition; increment )
{
    statement;
}
```

For loop

为了

```
for(int a = 0; a < 10; a++)
{
    // …
}
```

increment

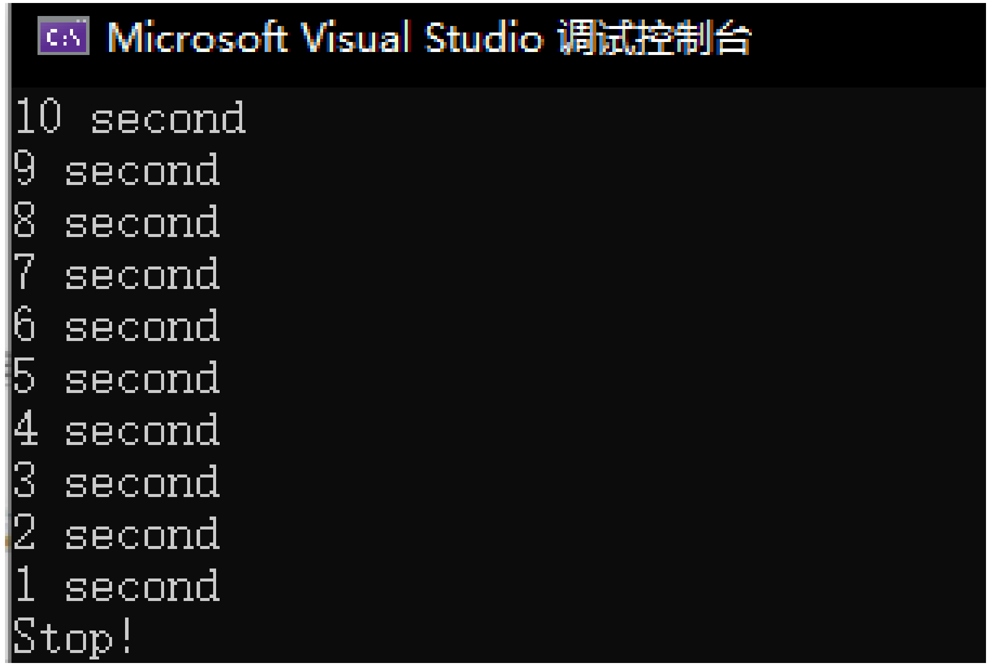
```
for(int a = 100; a >= 0; a--)
{
    // …
}
```

decrement

Case study: for loop

Case: how to make a counter?

```
#include <stdio.h>
main ()
{
     for(int sec = 10; sec>0; sec--)
     {
       printf("%d second\n", sec);
     }
     printf(“Stop!”);
}
```



```
10 second
9 second
8 second
7 second
6 second
5 second
4 second
3 second
2 second
1 second
Stop!
```

While loop

While loop repeatedly executes a statement as long as the condition is true.

```
while(condition)
{
     statement;
}
```

While loop

当

```
int a = 0;
while(a < 10)
{
     // …
    a++;
}
```

```
int a = 100;
while(a >= 0)
{
     // …
    a++;
}
```

For versus while

```
for(int a = 0; a < 10; a++)
{
    // …
}
```

Same

```
int a = 0;
while(a < 10)
{
    // …
    a ++;
}
```

```
for(int a = 100; a >= 0; a--)
{
    // …
}
```

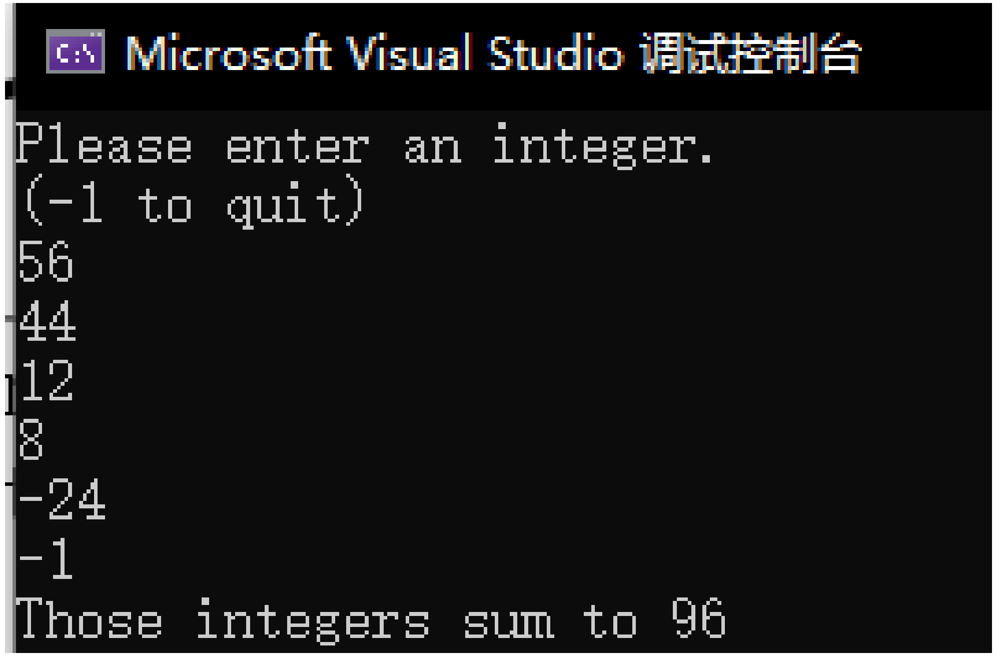
Same

```
int a = 100;
while(a >= 0)
{
    // …
    a --;
}
```

Case study: while loop

Case: sum the user’s input, exit when input -1.

```
#include <stdio.h>
main ()
{
    printf(“Enter an integer.\n(-1 to quit)\n");
    int input_num = 0;
    int sum = 0;
    while (input_num != -1)
    {
        scanf_s("%d", &input_num);
        sum = sum + input_num;
     }
     printf("Those integers sum to %d", sum);
}
```



```
Please enter an integer.
(-1 to quit)
56
44
12
8
-24
-1
Those integers sum to 96
```

Do-while loop

do-while loop is similar to while loop, it guarantees to execute at least one time.

```
do
{
    statement;
}while( condition );
```

Do-while loop

```
int a = 0;
while(a < 10)
{
    // …
   a++;
}
```

做

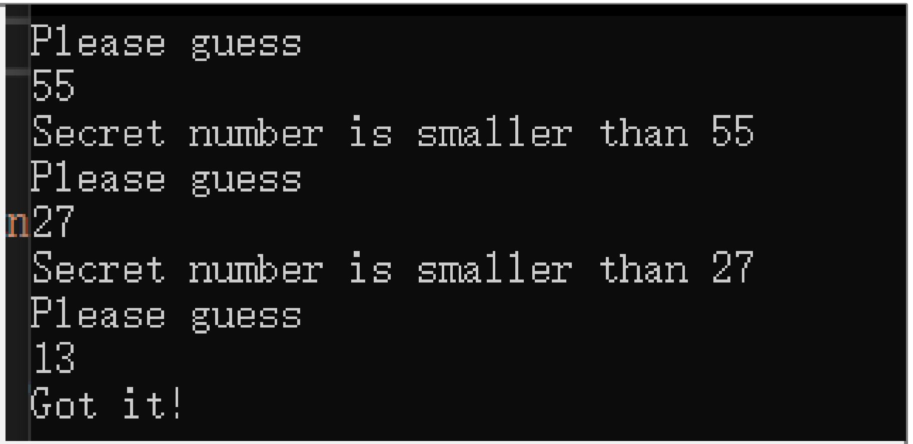
```
int a = 0;
do
{
    // …
   a++;
} while(a < 10)
```

当

Case study: do-while loop

Case: find the secrete number.

```
#include <stdio.h>
main ()
{
    int num;
    int secret_num = 13;
    do{
       printf("Please guess\n");
       scanf("%d", &num);
       if (num > secret_num) {
           printf("Secret number is smaller than %d\n", num);}
       if (num < secret_num) {
           printf("Secret number is larger than %d\n", num);}
    } while (secret_num!=num);
    printf("Got it!\n");
}
```



```
Please guess
55
Secret number is smaller than 55
Please guess
27
Secret number is smaller than 27
Please guess
13
Got it!
```

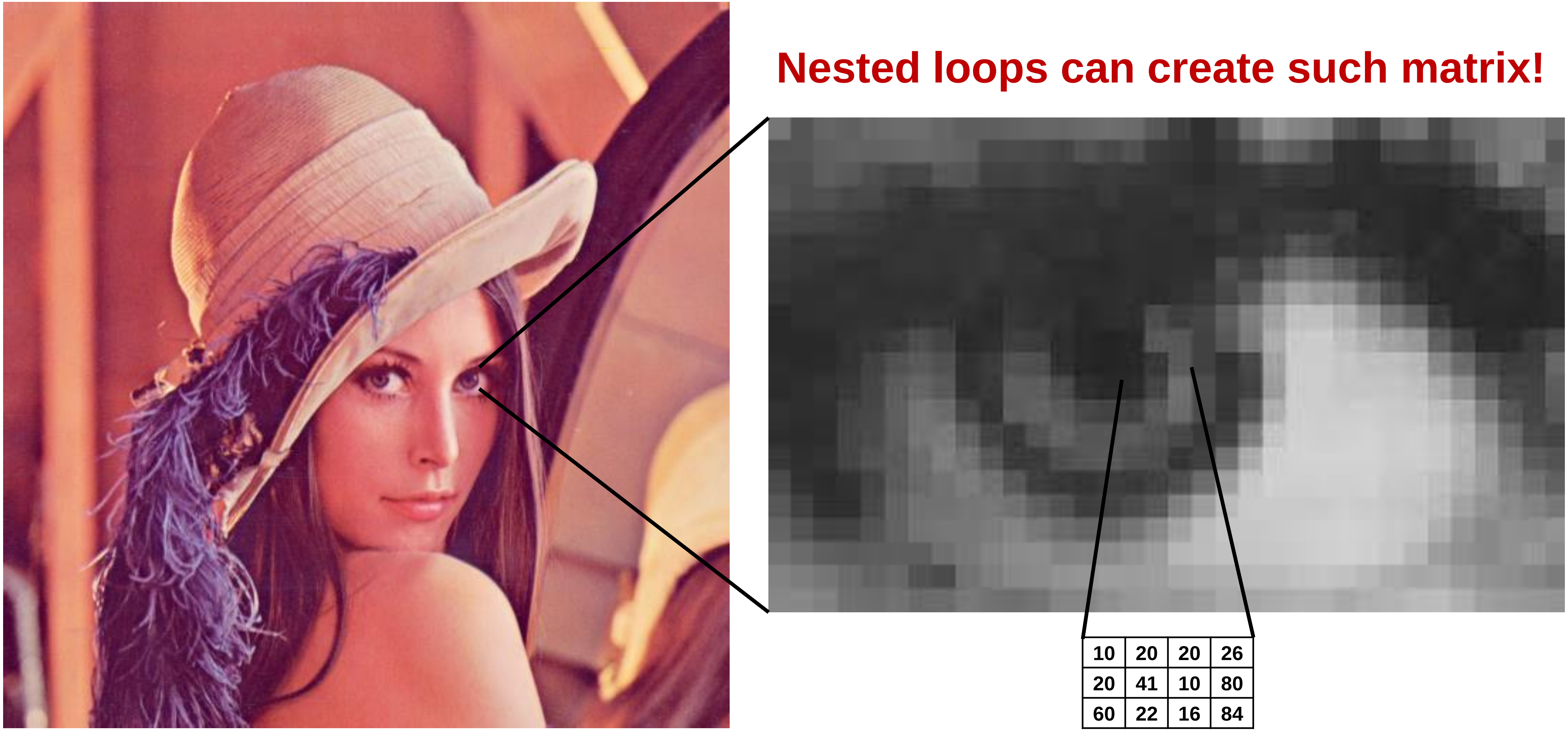
Nested loops

C allows using one loop inside another loop.

```
for ( init; condition; increment )
{
     for ( init; condition; increment )
     {
          // xxxx
     }
}
```

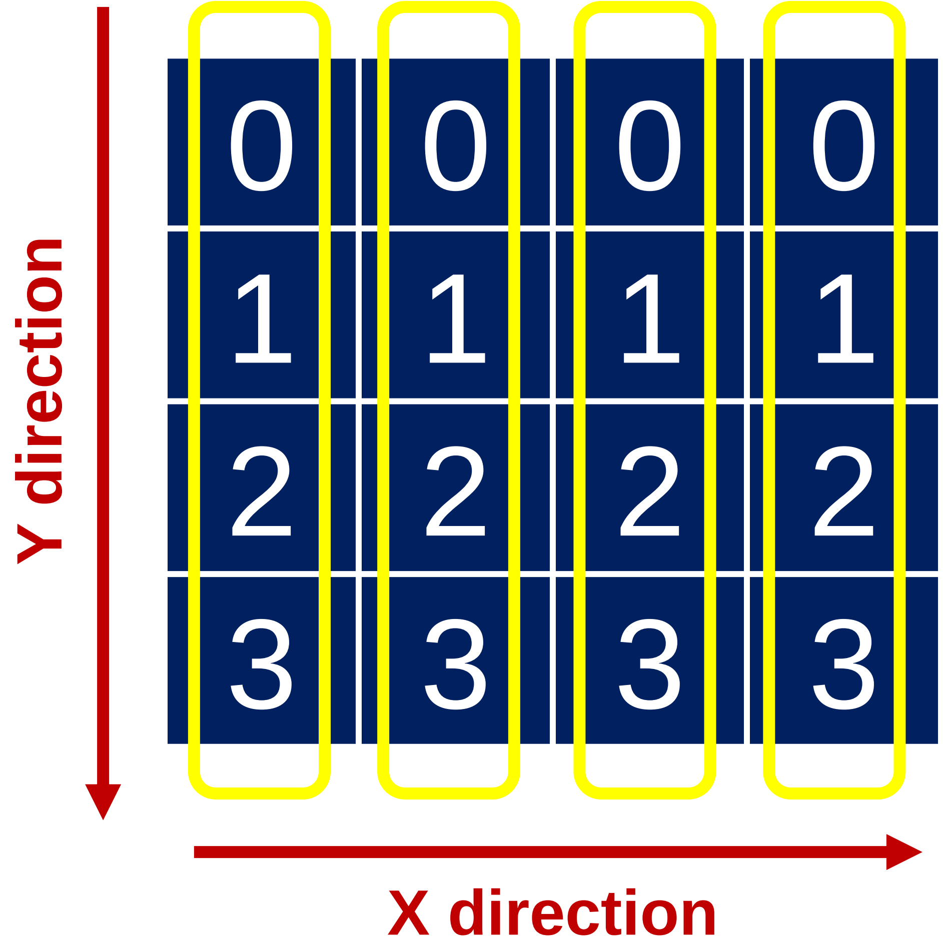
Nested loops

Nested loops can create such matrix!



10	20	20	26
20	41	10	80
60	22	16	84

Nested loops

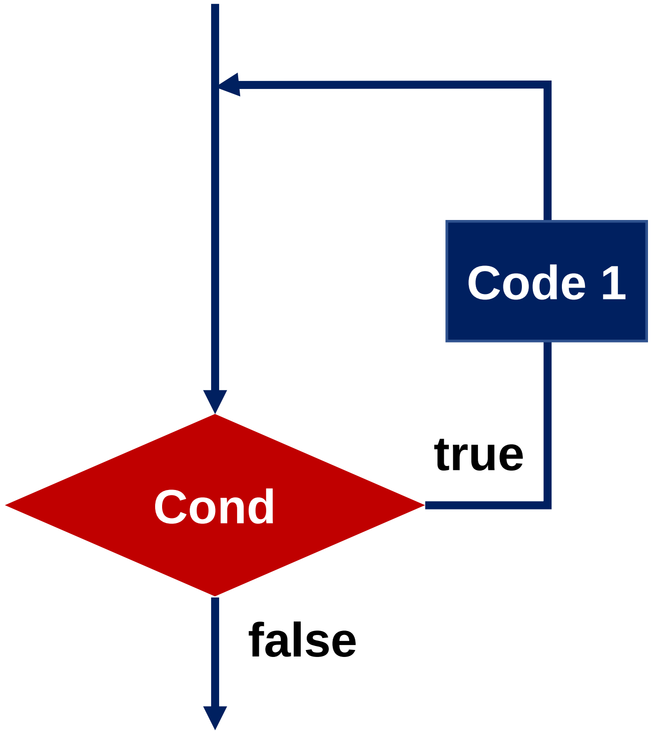


0	0	0	0
1	1	1	1
2	2	2	2
3	3	3	3

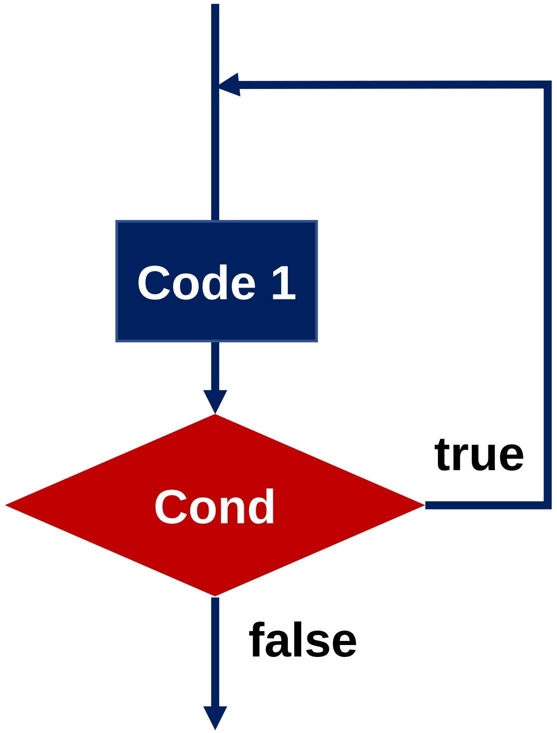
```
for (int x = 0; x < 4; x++)
{
    for (int y = 0; y < 4; y++)
    {
        // fill y at <x, y>
    }
}
```

Overview of loops

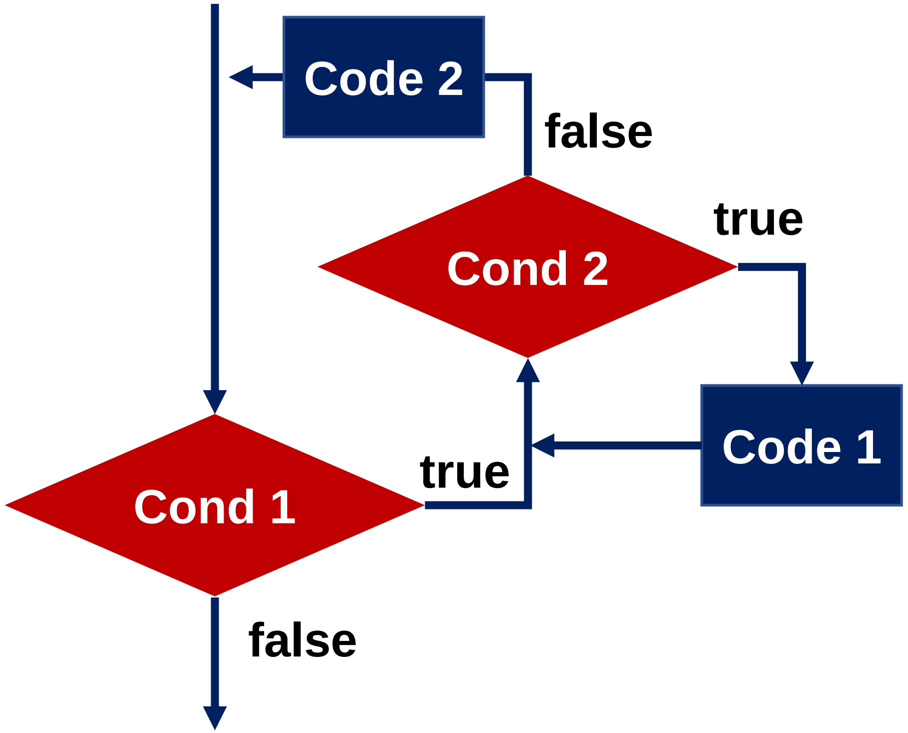
for/while loop



do-while loop



nested for loop



Same task in 3 looping formats

Calculate the sum = 1+2+..100

For loop

```c
#include <stdio.h>
int main()
{
    int sum = 0;
    for (int i=1;i<=100;i++)
   {
       sum += i;
    }
    printf(“sum=%d\n", sum);
    return 0;
}
```

While loop

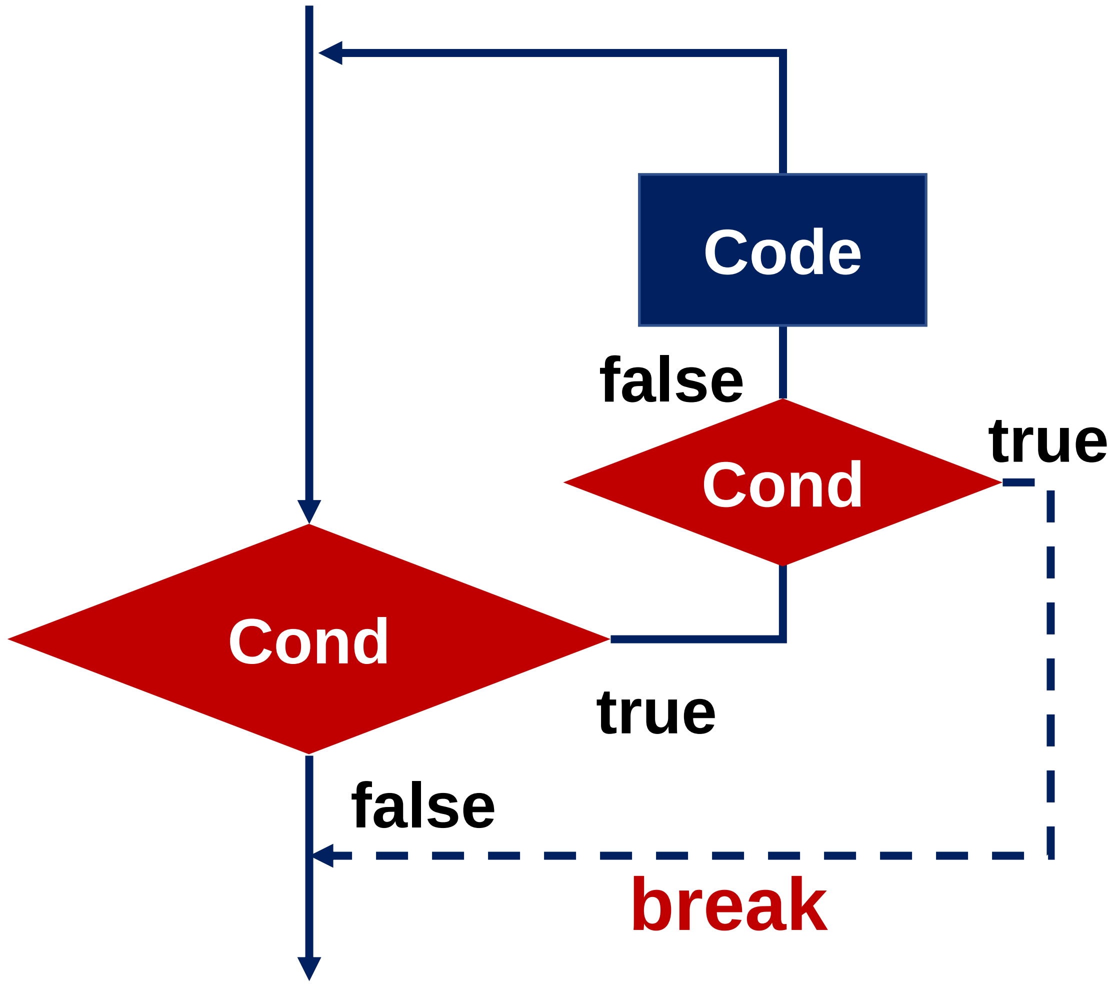
```c
#include <stdio.h>
int main()
{
    int sum = 0, i = 1;
    while(i <= 100) {
       sum += i;
       i++;
    }
    printf(“sum=%d\n", sum);
    return 0;
}
```

Do-while loop

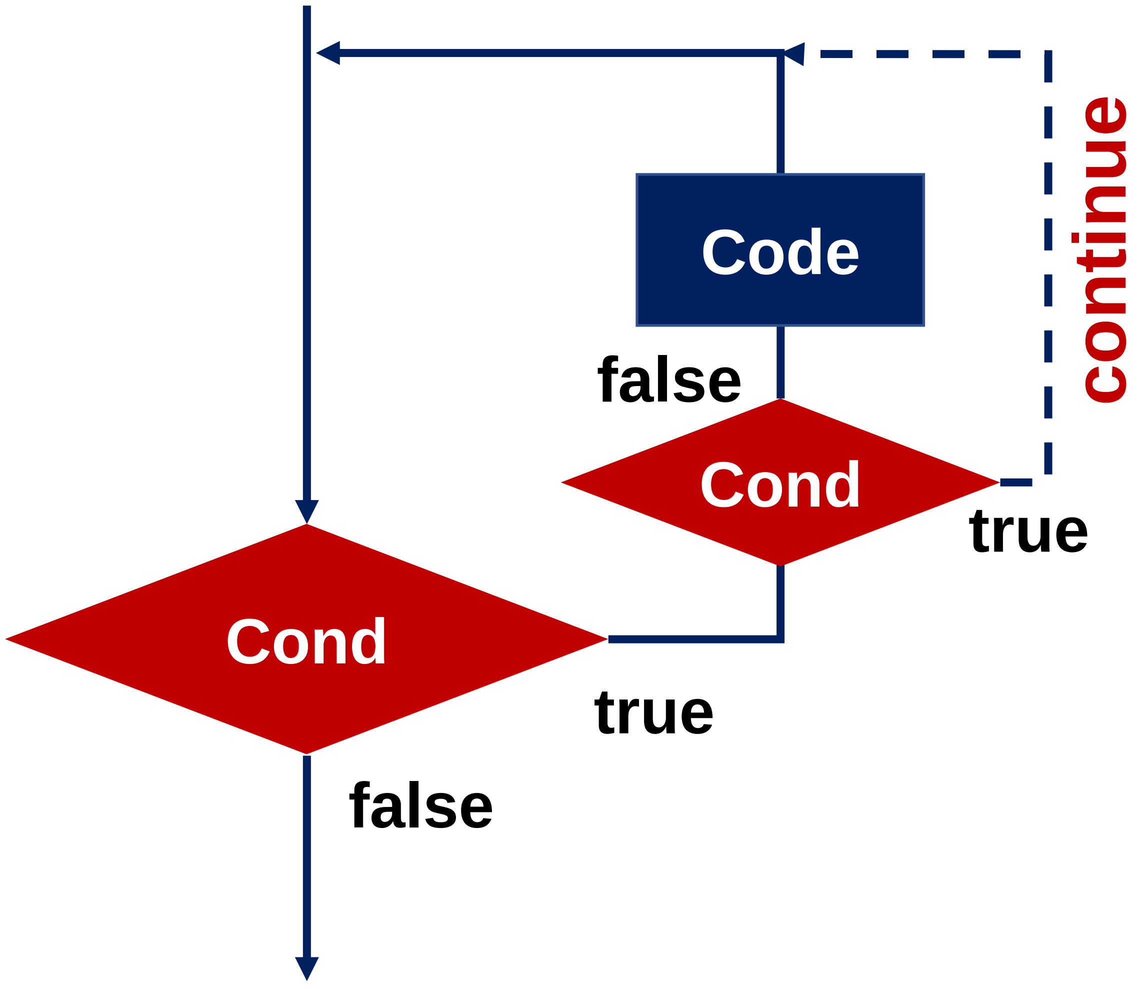
```c
#include <stdio.h>
int main()
{
    int sum = 0, i = 1;
    do {
       sum += i;
       i++;
    } while (i <= 100);
    printf(“sum=%d\n", sum);
    return 0;
}
```

Break and continue

Break loop



Continue loop



Break statement

Break terminates the loop when meeting the criterion.

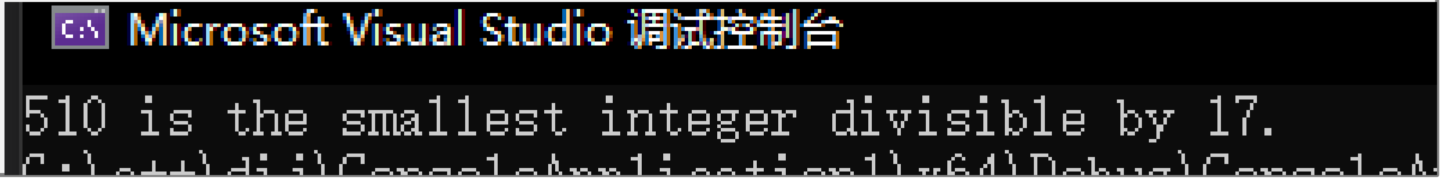
```
for ( init; condition; increment )
{
    if (statement)
        break;
}
```

Break is needed for brute-force searching!

Case study: break statement

Case: output the smallest integer divisible by 17 but greater than 500

```c
#include <stdio.h>
int main ()
{
    int num = 500;
    while (1){
        if (num % 17 == 0) {
            printf("%d is the smallest integer divisible by 17.", num);
            break;
        }
        num++;
    }
    return 0;
}
```



```
510 is the smallest integer divisible by 17.
C:\c++\diji\ConsoleApplication1\x64\Debug\ConsoleA
```

Continue statement

Continue forces execution to the next iteration, skipping the code in between.

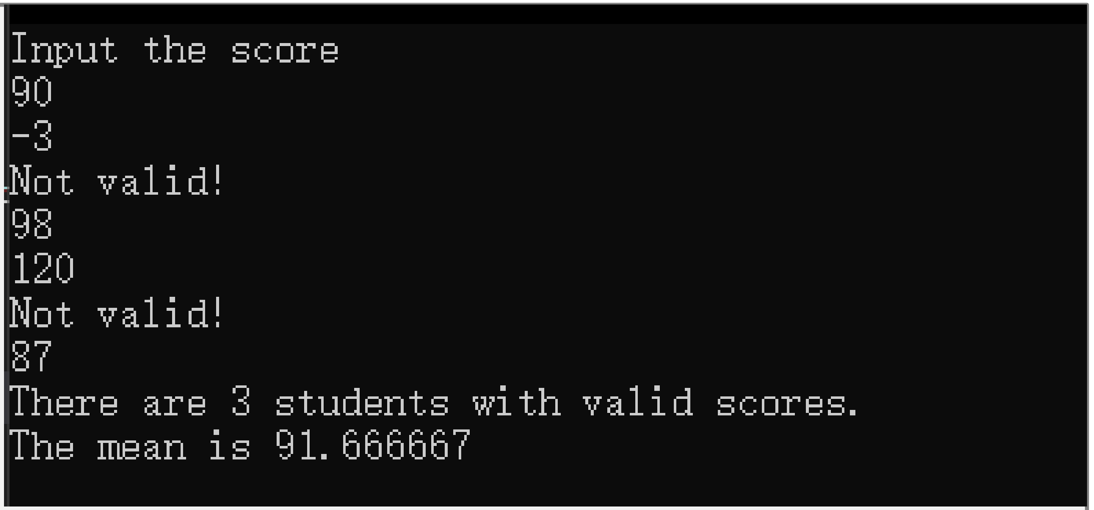
```
for ( init; condition; increment )
{
        if (condition)
                continue;
        // …
}
```

Continue can skip unwanted rounds in looping!

Case study: continue statement

Case: calculate the average score of 5 students with valid scores in [0, 100].

```
#include <stdio.h>
main ()
{
    int number = 0, scores = 0,sum =0;
    printf("Input the score\n");
    for (int i = 0;i < 5;i++) {
        scanf ("%d", &scores);
        if (scores < 0 || scores >100) {
            printf("Not valid!\n");
            continue;
        }
        number++;sum += scores;
    }
    printf("There are %d students with valid scores.\nThe mean is %f\n", number,sum * 1.0
/ number);
}
```



```
Input the score
90
-3
Not valid!
98
120
Not valid!
87
There are 3 students with valid scores.
The mean is 91.666667
```

Infinite loop - Virus!

NOTE: A loop becomes **infinite** if a condition never becomes **false**!

```
#include <stdio.h>

int main ()
{
    for( ; ; ) // while(true)
    {
        printf("This loop will run
        forever.\n");
    }
    return 0;
}
```

Summary

1. Decision-making (if, switch)
2. Looping (for, while)

Summary

- Two major workflow controls provided in C: **decision-making** and **looping**
- Two types of statement for making decisions: **if-else** and **switch**, if-else is more popular, switch is for equality check
- Two types of statement for looping: **for** loop and **while/do-while** loop, both are essentially the same
- **Break** and **continue** statements can be used to influence loops, jump out from the loop or skip specific loops
- Time to write you C program to control workflows

Homework

1. Enter a score (in range 0 to 100) and convert it to a corresponding grade. The conversion rules are as follows:

a) A: 90~100;
B: 80~89;
C: 70~79;
D: 60~69;
E: 0~59;

b) Enter the score with “scanf” and print the corresponding grade with “printf”

c) Test input: 95, 59

2. Enter an integer and print factorial of the number(n!).

a) Enter the integer with “scanf”

b) Test input :10

Homework

3. Find and print the prime numbers (integer) between 100 - 200.

a) A prime number is a natural number greater than 1 and it has no other factors except 1 and itself

4. Print a diagonal matrix with 4 rows and 4 cols, the value on the diagonal is equal to its row number.

a) Use nested loops (for-for) to print the matrix and specify the values on the diagonal using if statement

b) The matrix looks like

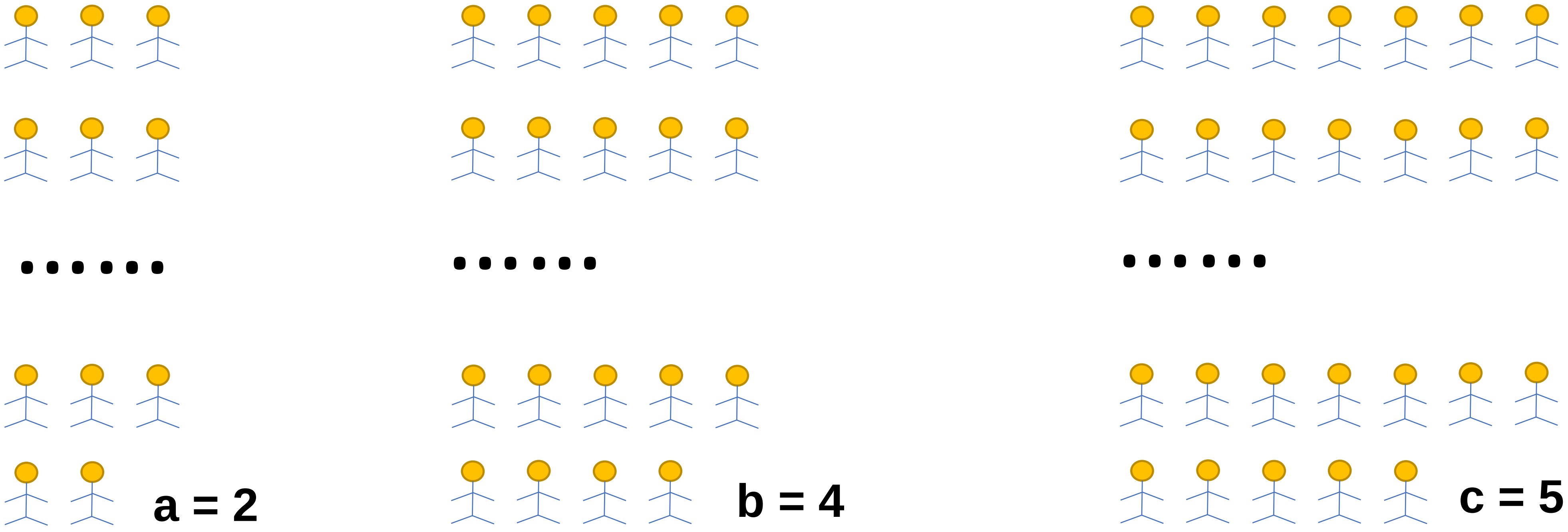
$$\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 4 \end{matrix}$$

Homework

5. Students are on the playground, grouping as 3 persons in a row, 5 in a row and 7 in a row, respectively. The number of students in the last row are a , b , c. How many students are on the playground ?

a) The total number of students is between 10 and 100

b) a = 2; b = 4; c = 5



Homework

6-A. (**bonus**) There is a pair of rabbits. From the third month after birth, a pair of rabbits are born every month. After the little rabbit grows to the third month, another pair of rabbits is born every month. Please print the total number of rabbits on each month.

a) The rabbits will die on the 6 month
b) You should print the number of rabbits for the first 20 months

6-B. (**bonus**) Watch one of the movies (about looping) mentioned in the lecture and write a 400 words review (in English).

Only choose one between 6-A and 6-B to get bonus!!!

Review of lecture 2 homework

? 的使用:

如何取a,b 中的较大值

```
#include <stdio.h>

int main()
{
        int a = 10;
        int b = 20;
        int c;
        c = a > b ? 1 : 0;
        printf("10 > 20 ? :%d\n", c);

        return 0;
}
```

```
#include <stdio.h>

int main()
{
        int a = 10;
        int b = 20;
        int c;
        c = a > b ? a : b;
        printf("10 > 20 ? :%d\n", c);

        return 0;
}
```

Review of lecture 2 homework

Scanf与scanf_s：

scanf是一个古老的函数，在读取字符时可能会发生内存错误，微软提供了全新的scanf_s函数，该函数在读取字符时需要输入字符的长度，避免了内存越界

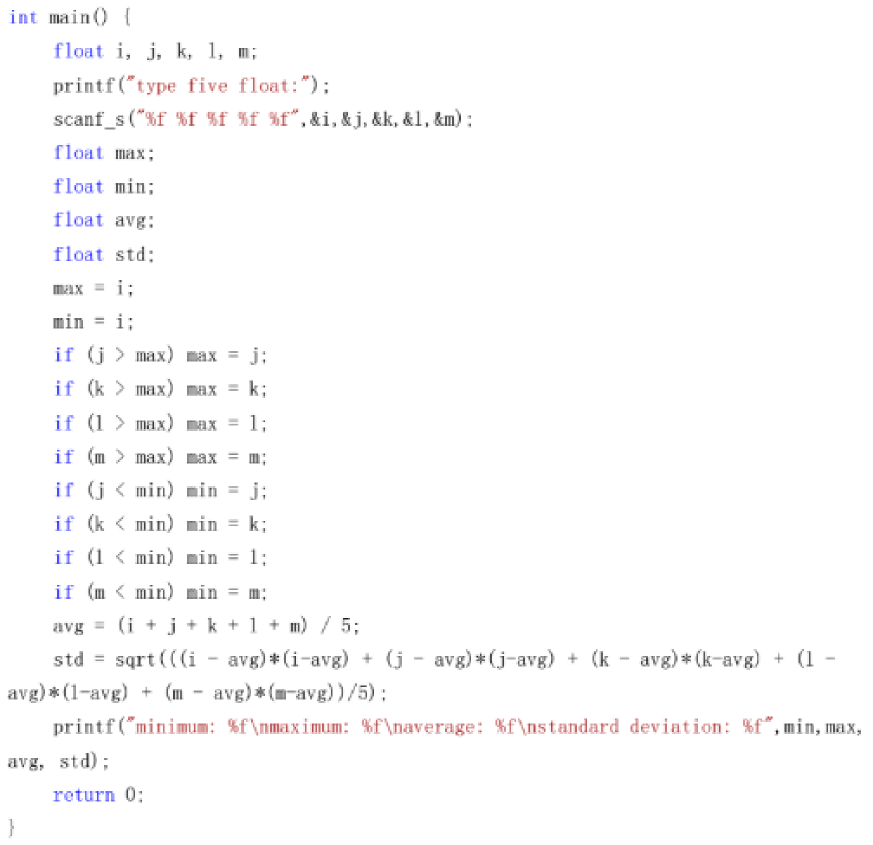
```
#define
_CRT_SECURE_NO_WARNINGS
#include<stdio.h>

int main()
{
int a;
char b;
float c;
scanf("%d,%c,%f",&a,&b,&c);
printf("%d %c %f",a,b,c);

}
```

```
#include<stdio.h>

int main()
{
int a;
char b;
float c;
scanf_s("%d,%c,%f",&a,&b,1,&c);
printf("%d %c %f",a,b,c);

}
```

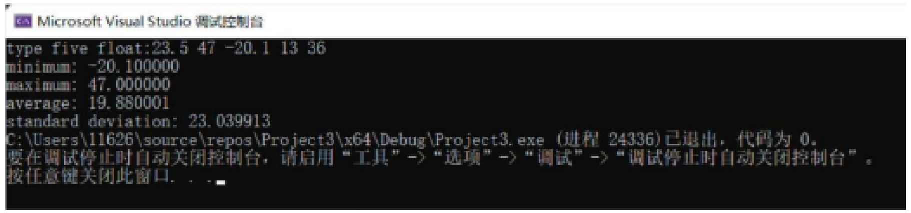
Review of lecture 2 homework

作业的格式：

PDF（源码文本 + 运行结果截图）



```
int main() {
    float i, j, k, l, m;
    printf("type five float:");
    scanf_s("%f %f %f %f %f",&i,&j,&k,&l,&m);
    float max;
    float min;
    float avg;
    float std;
    max = i;
    min = i;
    if (j > max) max = j;
    if (k > max) max = k;
    if (l > max) max = l;
    if (m > max) max = m;
    if (j < min) min = j;
    if (k < min) min = k;
    if (l < min) min = l;
    if (m < min) min = m;
    avg = (i + j + k + l + m) / 5;
    std = sqrt(((i - avg)*(i-avg) + (j - avg)*(j-avg) + (k - avg)*(k-avg) + (l -
avg)*(l-avg) + (m - avg)*(m-avg))/5);
    printf("minimum: %f\nmaximum: %f\naverage: %f\nstandard deviation: %f",min,max,
avg, std);
    return 0;
}
```



```
type five float:23.5 47 -20.1 13 36
minimum: -20.100000
maximum: 47.000000
average: 19.880001
standard deviation: 23.039913
C:\Users\11626\source\repos\Project3\x64\Debug\Project3.exe (进程 24336)已退出，代码为 0。
要在调试停止时自动关闭控制台，请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”。
按任意键关闭此窗口. . .
```