Loop

Tai, Wei Hsuan

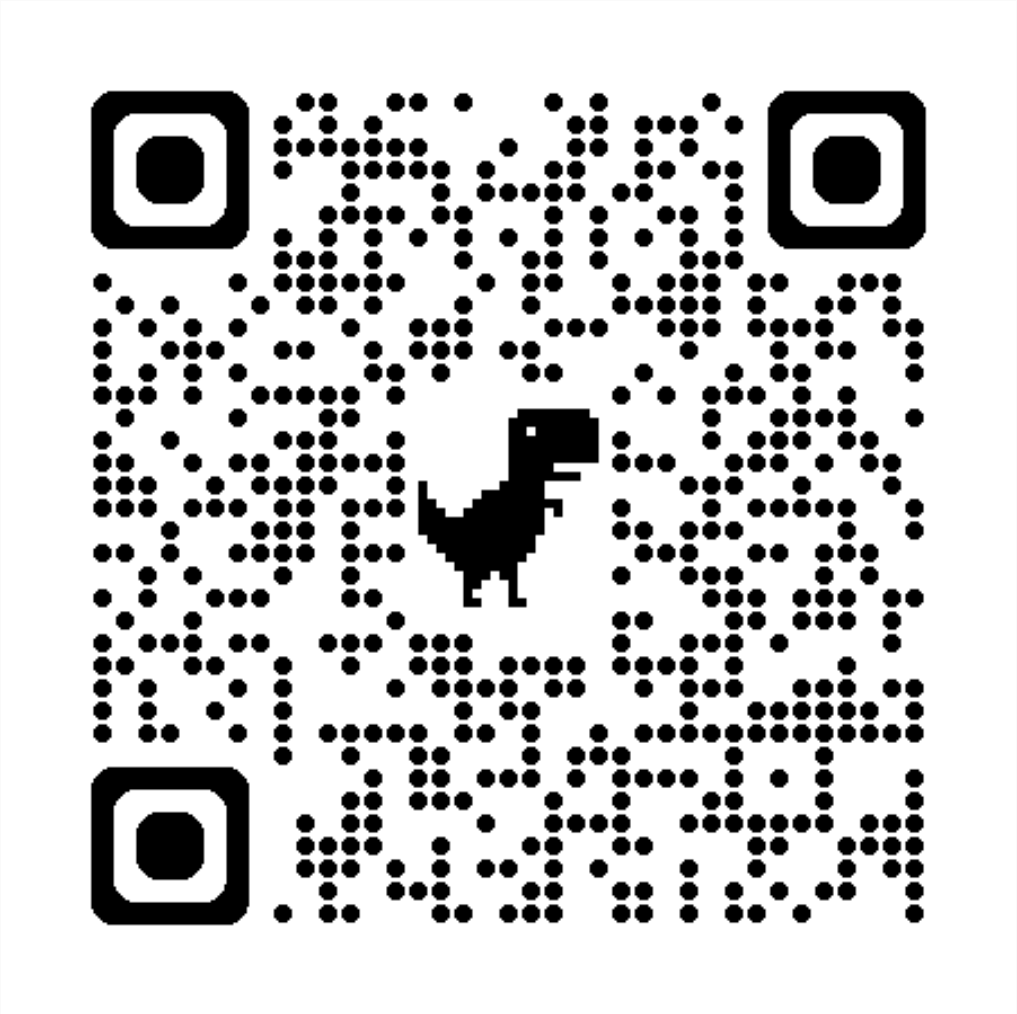
week 4

Announcement

Recording or photographing the slides or any content displayed on the screen is permitted for personal use only.

Redistribution, modification, or any commercial use of the materials is strictly prohibited.

Course Materials



Outline

1. Recap

2. Loop

Recap

Basic Structure

```
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  int main(){
5      // code here
6      return 0;
7  }
```

Variables and basic I/O

```
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  int main(){
5      int a;
6      cin >>a;
7      cout<<"The value of a is: "<<a<<'\n';
8  }
```

Selection Statements

```
1               if(condition){
2                   // code when condition is true
3               }else if(condition2){
4                   // code when condition2 is true
5               }else{
6                   // code when both conditions are false
7               }
```

Example: Leap Year Determination

According to the science state, the total time for the Earth to orbit the Sun is approximately 365.25 days. To account for this extra quarter day, an extra day is added to the calendar every four years, resulting in a leap year with 366 days. However, to further refine the calendar and maintain alignment with the solar year, additional rules are applied:

- It can devise by 4.
- It can not devise by 100, unless it can devise by 400.

For instance, the year 2000 is a leap year because it is divisible by 400, while the year 1900 is not a leap year because it is divisible by 100 but not by 400.

Strategy

While executing a determination operator, we should list the levels of conditions from the most general to the most specific, applying the rules from wild to strict (like a sieve), because the strict rule is a subset of the wild rule.

Though we can utilize `else if` to avoid redundant checks, but this strategy can also help us to think more clearly.

Strategy(cont.)

Take the leap year determination as an example, if you don't care the level of conditions, you may write the code like this:

```
1               if(a%100==0)ly=0;
2               else if(a%400==0)ly=1;
3               else if(a%4==0)ly=1;
```

What's the problem? What if the year is 2000?

Strategy(cont.)

In any case, you should list the conditions ranked by their levels ascendingly or descendingly, randomly mixing them is **CHAOS and EVIL!**

One possible correct code is:

```
1               if(a%400==0)ly=1;
2               else if(a%100==0)ly=0;
3               else if(a%4==0)ly=1;
```

Strategy(cont.)

If you list the level from wild to strict, you can even omit the `else` statements:

```
1               if(a%4==0)ly=1;
2               if(a%100==0)ly=0;
3               if(a%400==0)ly=1;
```

Loop

Basic Conception

If you want to repeat a block of code multiple times, you can use loops. Loops are used to execute a block of code repeatedly until a certain condition is met.

You can also skip or terminate the loop based on certain conditions using `continue` and `break` statements.

Two Types of Loops

There are two primary types of loops in C++:

- **For Loop**: Used when the number of iterations is known beforehand.
- **While Loop**: Used when the number of iterations is not known beforehand and depends on a condition.

In one word, if you know how many times you want to repeat, use `for`, otherwise use `while`.

For loop

In the initialization, you can declare and initialize loop control variables that would only be used in the loop. The condition is checked before each iteration, and if it evaluates to true, the loop body is executed. After each iteration, the update statement is executed to modify the loop control variables.

Remember the variables' scope rules! Variables declared in the initialization part are only accessible within the loop.

```
for(initialization; condition; update){
    // code to be executed
}
```

Example

For instance, to print numbers from 1 to 10:

```
for(int i=1;i<=10;i++){
    cout<<i<<'\n';
}
cout<<i<<'\n'; // Error: i is not defined here
```

One common usage of `for` loop is to iterate through arrays or collections.

```
for(int i=0;i<n;i++){
    cout<<arr[i]<<'\n';
}
```

While loop

The `while` loop continues to execute the block of code as long as the specified condition is true. The condition is checked before each iteration, and if it evaluates to false, the loop terminates.

```
while(condition){
    // code to be executed
}
```

Example

For example, to print numbers from 1 to 10 using a `while` loop:

```
1               int i=1;
2               while(i<=10){
3                   cout<<i<<'\n';
4                   i++;
5               }
```

But we often use `for` loop when the number of iterations is known.

Input until EOF(End of File)

In competitive programming, you often need to read input until the end of the file (EOF). This can be done using a `while` loop in combination with the `cin` stream.

```
int x;
while(cin>>x){
    // process x
}
```

Some tricks

If you are not sure when the loop should end, you can use an infinite loop with a `break` statement to exit the loop when a certain condition is met.

```
1            while(1){
2                // code to be executed
3                if(condition_to_exit){
4                    break;
5                }
6            }
```

Some tricks

If you want to skip the current iteration and move to the next one, you can use the `continue` statement.

```
while(1){
    // code to be executed
    if(condition_to_exit){
        continue;
    }
}
```

Note that `break` and `continue` can be used in both `for` and `while` loops, but they only affect the innermost loop they are placed in.

Example

Here is an example of using `continue` in a loop to show only odd numbers from 0 to 9:

```
1               for(int i=0;i<10;i++){
2                   if(i%2==0)continue;
3                   cout<<i<<'\n';
4               }
```

Common issue

This code snippet demonstrates the use of `break` inside nested loops. The `break` statement only exits the innermost loop, not all loops.

```
1             for(int i=0;i<10;i++){
2                 for(int j=0;j<10;j++){
3                     if(j==5)break;
4                     cout<<i<<' '<<j<<'\n';
5                 }
6             }
```