- What can be done if no value can be chosen as sentinel ?

  - You might have to enter an extra value in each iteration

  - The only purpose of this value will be to signal the end of the data

```
// Computes the sum of a list of numbers
#include <iostream.h>

void main()
{
   int i, num,sum;

   cout << "Enter a number \n";
   cin >> num;
   sum=0;
   while(num != -9999)
   {
      sum = sum + num;
      cout << "Enter a number \n";
      cin >> num;
   }
   cout << "The sum is" << sum << endl;
}
```

```
// Computes the sum of a list of numbers
#include <iostream.h>

void main()
{
   int i, num,sum,cont;

   cout << "Enter a number and 1 (continue) or 0 (stop)\n";
   cin >> num  >> cont;
   sum=0;
   while(cont == 1)
   {
      sum = sum + num;
      cout << "Enter a number and 1 (continue) or 0 (stop)\n";
      cin >> num >> cont;
   }
   cout << "The sum is" << sum << endl;
}
```

an extra value is used to indicate if there is more data or not

- End-of-file-controlled loops

  - When data is read from a file, the EOF (end-of-file) can be detected and used as a sentinel

  - How can we detect the EOF ?

    * When the program attempts to read data beyond the EOF, the input stream enters the *fail state*

    * Thus, EOF can be detected by simply detecting the fail state !!

    * *Warning:* there are other cases which can also cause the fail state ...

```
#include <iostream.h>
#include <fstream.h>

void main()
{
 int sum, num;
 ifstream fin;

 fin.open("myfile.dat");

 sum=0;
 fin >> num; // priming read

 while(fin != 0) // i.e., while EOF has not been reached
 {
  sum = sum + num;
  fin >> num;       // get the next value
 }
 cout << "The sum is " << sum << endl;
}
```

- Can the same strategy be used when data is entered using the keyboard ?

- Yes !!! (the character *CtrlZ* can be used to signify EOF in this case)

```
#include <iostream.h>

void main()
{
 int sum, num;

 sum=0;
 cin >> num; // priming read

 while(cin != 0) // i.e., while EOF has not been entered (CtrlZ)
 {
  sum = sum + num;
  cin >> num;       // get the next value
 }
 cout << "The sum is " << sum << endl;
}
```

- Flag-controlled loops

  - Boolean variables (*flags*) can be used in the test condition of the while loop

```
#include <iostream.h>
typedef int Boolean;

void main()
{
 int sum, num;
 Boolean nonNegative;

 sum=0;
 cin >> num; // priming read

 nonNegative=1; // it is initialized to TRUE
 while(nonNegative != 0) // i.e., while TRUE
 {
  sum = sum + num;
  cin >> num;      // get the next value

  if(num < 0)      // if num is negative, nonNegative becomes FALSE
    nonNegative=0;
 }
 cout << "The sum is " << sum << endl;
}
```

- Designing loops

  - Initialization, Process, Update

- Examples

  1. Print the numbers from 1 to 10 (write test condition in a different form if possible)

  2. Print the numbers from 10 to 1 (write test condition in a different form if possible)

  3. Print a table of numbers from 1 to 10 with their squares and cubes

  4. The formula for converting Celsius to Fahrenheit temperatures is *Fahrenheit=(9/5) Celsius+32*. Write a program which prints Celsius temperatures from 5 to 50 degrees and their corresponding Fahrenheit temperatures in increments of five degrees.

  5. Rewrite the above program so that it starts at a Celsius value of -10 and ends with a Celsius value of 60, in increments of 10 degrees.

6. Modify the above program to request the starting Celsius value, the ending Celsius value, and the increment.

7. Write a program that reads a list of numbers and prints out the largest

8. An algorithm for a company payroll (see Chapter 1, page 32)

a) *Data for each employee:* identification number, the hourly pay rate, and the hours worked that week

b) *Wages:* If an employee has worked at most 40 hours, multiply the employee's pay rate with the number of hours. If an employee has worked more than 40 hours, multiply the employee's pay rate with 40, plus one and a half times the employee's regular pay rate with the number of hours worked above 40.

c) The results should be stored in a file called *payFile*

d) There is no employee number 0, so the user can indicate the end of the data by entering a 0 when asked for an employee's number.

- Solutions to the examples

```
#include <iostream.h>
void main()
{
 int i;

 i=1;
 while (i <= 10)  // or (i < 11)
 {
  cout << i;
  ++i;
 }
}
```

```
#include <iostream.h>
void main()
{
 int i;

 i=10;
 while (i >= 0)   // or (i > -1)
 {
  cout << i;
  --i;
 }
}
```

```
#include <iostream.h>
#include <iomanip.h>
void main()
{
 int i, square, cube;

 i=1;
 while (i <= 10)
 {
  square = i * i;
  cube = i * i * i;
  cout << setw(6) << i << setw(6) << square << setw(6) << cube << endl;
  ++i;
 }
}

#include <iostream.h>
void main()
{
 int celsius;
 float fahren;

 celsius = 5;
 while (celsius <= 50)
 {
  fahren = (9.0/5.0) * celsius + 32.0;
  cout << "Celsius: " << celsius << " Fahrenheit: " << fahren << endl;
  celsius = celsius + 5;
 }
}

#include <iostream.h>
void main()
{
 int celsius;
 float fahren;

 celsius = -10;
 while (celsius <= 60)
 {
  fahren = (9.0/5.0) * celsius + 32.0;
  cout << "Celsius: " << celsius << " Fahrenheit: " << fahren << endl;
  celsius = celsius + 10;
 }
}
```

```
#include <iostream.h>
void main()
{
 int celsius, start, end, incr;
 float fahren;

 cout << "Enter starting and ending Celsius values: ";
 cin >> start >> end;
 cout << "Enter increment: ";
 cin >> incr;

 celsius = start;
 while (celsius <= end)
 {
  fahren = (9.0/5.0) * celsius + 32.0;
  cout << "Celsius: " << celsius << " Fahrenheit: " << fahren << endl;
  celsius = celsius + incr;
 }
}
```

```
#include <iostream.h>
int main()
{
 float num, largest;

 cout << "Enter a number or Ctrl Z to end: ";  // priming read
 cin >> num;

 if(cin != 0)
   largest=num; // initialize largest
 else
 {
   cout << "No data entered !!!" << endl;
   return 1;
 }

 while(cin != 0)
 {
  cout << "Enter a number or Ctrl Z to end: ";
  cin >> num;
  if(num > largest)
    largest = num;
 }
 cout << "The largest number entered is: " << largest << endl;

 return 0;
}
```

```
#include <iostream.h>
#include <fstream.h>

void main()
{
 int idNumber, hours;
 float payRate, wage;
 ofstream fout;

 fout.open("payFile");

 cout << "Enter employee's number or 0 to end: ";
 cin >> idNumber;
 while(idNumber < 0)  // test for invalid id
 {
   cout << "Enter employee's number or 0 to end: ";
   cin >> idNumber;
 }

 while(idNumber != 0)  // main while loop
 {
   cout << "Enter employee's pay rate: ";
   cin >> payRate;
   while(payRate < 0)  // test for invalid pay rate
   {
     cout << "Enter employee's pay rate: ";
     cin >> payRate;
   }

   cout << "Enter number of hours: ";
   cin >> hours;
   while(hours < 0)
   {
     cout << "Enter number of hours: ";  //test for invalid number of hours
     cin >> hours;
   }

   if(hours <= 40)
     wage = payRate * hours;
   else
     wage = (payRate * 40) + (1.5 * payRate * (hours - 40));

   fout << "Employee:" << idNumber << " wage: " << wage << endl;

   cout << "Enter employee's number or 0 to end";
   cin >> idNumber;
   while(idNumber < 0)  // test for invalid id
   {
     cout << "Enter employee's number or 0 to end";
     cin >> idNumber;
   }
 }
}
```

- Keeping track of a previous value

- Sometimes, we want to remember the previous value of a variable

Find the number of times the pair "++" appears in a setence


```
#include <iostream.h>

void main()
{
 int count;
 char prev_ch, cur_ch;

 count=0;

 cin.get(prev_ch);
 while(prev_ch != '\n')  // Sentinel-controlled loop
 {
  cin.get(cur_ch);
  if((cur_ch == '+') && (prev_ch == '+'))
    ++count;
  prev_ch=cur_ch;
 }
 cout << "The pair ++ appears " << count << " time(s)" << endl;
}
```

- Nested logic

  - The body of a *while* or *if* statement can contain other *while* or *if* statements

  - Nested *while* statements

```
Initialize outer loop
while (outer loop condition)
{
Initialize inner loop
while (inner loop condition)
{
 Inner loop processing
 Inner loop update
}
Outer loop update
}
```

  - Rewrite the previous program to examine more than one lines

```
#include <iostream.h>

void main()
{
 int count;
 char prev_ch, cur_ch;

 count=0;

 cin.get(prev_ch);  // initialize outer loop
 while(cin != 0)    // test outer loop's condition -- EOF controlled loop
 {
   cin.get(cur_ch); // initialize inner loop
   while(cur_ch != '\n')   // test inner loop's condition -- Sentinel controlled loop
   {
     if((cur_ch == '+') && (prev_ch == '+'))
       ++count;
     prev_ch=cur_ch;
     cin.get(cur_ch);  // update inner loop
   }
   cin.get(prev_ch); // update outer loop
 }
 cout << "The pair ++ appears " << count << " time(s)" << endl;
}
```