Reading from a file

To read data from a file in C++, you can use the **std::ifstream** class from the **<fstream>** header. **std::ifstream** provides member functions to open a file, read data from the file, and close the file.

Here's an example of how to use **std::ifstream** to read a file line by line:

```
#include <fstream>
#include <iostream>
#include <string>

int main()
{
    std::ifstream file("input.txt");
    if (file.is_open())
    {
        std::string line;
        while (std::getline(file, line))
        {
            std::cout << line << std::endl;
        }
        file.close();
    }
    else
    {
        std::cout << "Unable to open file" << std::endl;
    }
    return 0;
}
```

In this example, the **std::ifstream** object **file** is used to open the file "input.txt". If the file is successfully opened, the **while** loop reads each line of the file using the **std::getline** function. The **std::getline** function reads a line of text from the input stream and stores it in a **std::string** object. The **std::endl** stream manipulator is used to insert a newline character after each line.

Once all the lines have been read, the **file.close()** function is called to close the file. If the file cannot be opened, an error message is printed to the console.

You can also use the >> operator to read data from the file. For example:

```
#include <fstream>
#include <iostream>

int main()
{
    std::ifstream file("input.txt");
    if (file.is_open())
    {
        int x;
        while (file >> x)
        {
            std::cout << x << std::endl;
        }
        file.close();
    }
    else
    {
        std::cout << "Unable to open file" << std::endl;
    }
    return 0;
}
```

In this example, the **while** loop reads integers from the file using the >> operator. The loop terminates when there are no more integers to read.

You can also use **std::ifstream** to read binary data from a file. To do this, you can use the **read** function, which reads a block of data from the file and stores it in a memory buffer.

Here's an example of how to read binary data from a file:

```
#include <fstream>
#include <iostream>

int main()
{
    std::ifstream file("input.bin", std::ios::binary);
    if (file.is_open())
    {
        char buffer[1024];
        while (file.read(buffer, sizeof(buffer)))
        {
            std::cout.write(buffer, sizeof(buffer));
        }
        file.close();
    }
    else
    {
        std::cout << "Unable to open file" << std::endl;
    }
```