Assignment 5

1. What are streams in C++ and why are they important?

Streams in C++ are abstractions that represent sources and destinations of data, such as input from the keyboard or output to the screen. They are important because they provide a consistent way to perform I/O operations.

2. Explain the different types of streams in C++.

- **Input Stream (istream)**: For reading input.
- **Output Stream (ostream)**: For writing output.
- **File Streams (ifstream, ofstream, fstream)**: For file input/output.
- **String Streams (istringstream, ostringstream, stringstream)**: For working with strings as streams.

3. How do input and output streams differ in C++?

- **Input streams (istream)** read data *into* a program.
- **Output streams (ostream)** send data *out* of a program.

4. Describe the role of the iostream library in C++.

The iostream library provides standard input/output stream classes like cin, cout, cerr, and clog.

5. What is the difference between a stream and a file stream?

- A **stream** handles general I/O.
- A **file stream** specifically handles reading from and writing to files.

6. What is the purpose of the cin object in C++?

cin is used to take input from the standard input device (keyboard).

7. How does the cin object handle input operations?

cin uses the extraction operator (>>) to take input and stores it in variables.

cpp

CopyEdit

int x;

cin >> x;

8. What is the purpose of the cout object in C++?

cout is used to display output to the standard output device (console).

9. How does the cout object handle output operations?

cout uses the insertion operator (<<) to send data to the console.

cpp

CopyEdit

cout << "Hello, World!";

10. Explain the use of the insertion (<<) and extraction (>>) operators.

- <<: Inserts output into ostream (e.g., cout << x)
- >>: Extracts input from istream (e.g., cin >> x)

11. What are the main C++ stream classes and their purposes?

- istream: Input stream
- ostream: Output stream
- ifstream: File input
- ofstream: File output
- fstream: File input/output
- stringstream: String manipulation via stream interface

12. Explain the hierarchy of C++ stream classes.

markdown

CopyEdit

ios

istream

ifstream

ostream

| └── ofstream

└── iostream

  └── fstream

13. What is the role of the istream and ostream classes?

- istream: Base class for all input streams.
- ostream: Base class for all output streams.

14. Describe the functionality of the ifstream and ofstream classes.

- ifstream: Reads data from files.
- ofstream: Writes data to files.

15. How do the fstream and stringstream classes differ from other stream classes?

- fstream: Supports both input and output on files.
- stringstream: Performs I/O operations on string objects instead of files or console.

16. What is unformatted I/O in C++?

Unformatted I/O reads or writes data as raw bytes or characters without formatting.

17. Provide examples of unformatted I/O functions.

cpp

CopyEdit

cin.get(ch);

cin.getline(str, size);

cout.put(ch);

18. What is formatted I/O in C++?

Formatted I/O provides control over how data is displayed (width, precision, etc.).

19. How do you use manipulators to perform formatted I/O in C++?

Using manipulators like setw, setprecision, fixed, etc.

cpp

CopyEdit

```
cout << setw(10) << fixed << setprecision(2) << value;
```

20. Explain the difference between unformatted and formatted I/O operations.

- **Unformatted**: Raw byte/character data.
- **Formatted**: Data with specific formatting (width, precision, etc.).

21. What are manipulators in C++?

Manipulators are functions that modify I/O stream behavior.

22. How do manipulators modify the behavior of I/O operations?

They set flags or properties on streams (e.g., number formatting, alignment).

23. Provide examples of commonly used manipulators in C++.

- setw(n)
- setprecision(n)
- fixed
- left, right
- endl

24. Explain the use of the setw, setprecision, and fixed manipulators.

- setw(n): Sets width of output field.
- setprecision(n): Sets decimal precision.
- fixed: Shows decimal point with precision.

25. How do you create custom manipulators in C++?

cpp

CopyEdit

```
ostream& custom(ostream& os) {
   os << "***";
   return os;
```

}

cout << custom << "Hello";

26. What is a file stream in C++ and how is it used?

A file stream is used for file I/O. Use ifstream, ofstream, or fstream.

27. Explain the process of opening and closing files using file streams.

cpp

CopyEdit

ifstream inFile("input.txt");

ofstream outFile("output.txt");

// Do file operations

inFile.close();

outFile.close();

28. Describe the different modes in which a file can be opened.

- ios::in – Read
- ios::out – Write
- ios::app – Append
- ios::trunc – Truncate
- ios::binary – Binary mode

29. How do you read from and write to files using file streams?

cpp

CopyEdit

ifstream fin("data.txt");

string name;

fin >> name;

ofstream fout("output.txt");

fout << "Name: " << name;