**1.** When dealing with input and output, how is the variety of devices dealt with in most modern computers?

Each input device has its own driver. Programs access these drivers through the i/o library, which makes the whole access for various input/output sources as similar as possible.

**2.** What, fundamentally, does an istream do?

Converts sequence of characters to values of different types and gets it from particular source (such as keyboard, file, memory etc.)

**3.** What, fundamentally, does an ostream do?

Converts values of different types to sequence of characters and sends it “somewhere” (keyboard, file, memory etc.)

**4.** What, fundamentally, is a file?

A sequence of bytes

**5.** What is a file format?

Type identifies how bytes in file should be interpreted

**6.** Name four different types of devices that can require I/O for a program.

Display, keyboard, microphone, sound system, internet connection

**7.** What are the four steps for reading a file?

a.) Know its name

b.) Open it

c.) Read in the characters

d.) Close

**8.** What are the four steps for writing a file?

a.) Name it

b.) Open it or create a new file of that name

c.) Write out the objects

d.) Close

**9.** Name and define the four stream states.

good() - The operation was successful

eof() - The end of file was hitted

fail() - Some error was encountered (e.g the type isn’t valid)

bad() - Something serious happened (e.g. disk read error)

**10.** Discuss how the following input problems can be resolved:

**a.** The user typing an out-of-range value

The program should output error message stating that the entered value doesn’t pass to the range and the state of stream should be changed to good(), so that user could continue to input values.

**b.** Getting no value (end of file)

This is how we often want to end getting the values so we simply return from a function.

**c.** The user typing something of the wrong type

If we declare special symbol for termination then we need to clear the state of stream and start searching for this symbol. If no symbol of particular type was found, the value should be returned to stream if another part of program uses it. In other cases we can simply terminate the input. Another solution is to set stream state in “good()” and continue to process values.

**11.** In what way is input usually harder than output?

The program needs to process values according to format of input, providing reasonable error-messages when the format is broken.

**12.** In what way is output usually harder than input?

We need to specify the format of output.

**13.** Why do we (often) want to separate input and output from computation?

If we run the program that relies on iostream, but at the moment streams fail to start, there is no possible benefit in running this program. That’s why we shouldn’t run it if there is no file to read or to write.

**14.** What are the two most common uses of the istream member function clear()?

1.) To recover stream from bad() or fail() state in order to search for some particular value (e.g. terminator)

2.) To skip values of particular type in stream (e.g. to skip all chars and leave only ints in stream)

**15.** What are the usual function declarations for << and >> for a user-defined type X?

ostream& operator<< (ostream&a, X x);

istream& operator>> (istream&a, X x);