Binary Files Solutions

Binary Files

- Write down a statement that opens an output file stream in binary mode

```
ofile.open("image.bmp", fstream:: binary);          // Open file for writing in binary mode
```

- Why are the << and >> operators not suitable for use with binary files?
  - Numeric data is converted to text
  - Text data is formatted on output
  - Whitespace characters are removed on input
- What should we use instead of these operators?
  - We should always use write() and read() with binary files

Binary File Structure

- Binary files usually have some kind of internal structure which makes the data meaningful
- If we need to work with a binary file which has a particular format, such as .jpg or .zip, what is the best way to achieve this?
  - The best way to work with a binary file is to create a struct whose data members correspond to the fields in the file format

Memory Alignment

- What does it mean for data to be "word aligned"?
    - An object, or a member of an object, is word aligned when its address is a multiple of the word size
    - e.g. on a 32-bit system, the address is a multiple of 4
- Why is word alignment important?
    - On modern computers, accessing data is much faster if the data is word aligned
    - Some architectures can only access data which is word aligned

Padding

- What is meant by "padding"?
  - If we have a struct or a class, an optimizing compiler will arrange the object in memory in the most efficient way possible
  - This may involve putting extra unused bytes between some of the struct members, so they can be accessed more quickly
  - These bytes are known as "padding" bytes
- Why does padding cause problems when working with binary files?
  - The data structure in memory must exactly match the file format
  - The padding bytes introduce a discrepancy between the two
  - Write operations will create an invalid file
  - Read operations will result in incorrect data values

#pragma pack

- What does #pragma pack mean?
  - #pragma pack is a non-standard compiler directive
  - However, it is supported by all the main compilers
  - It is used to set the data alignment between elements in a struct or class
- Why is #pragma pack helpful when working with binary files?
  - We can use #pragma pack to prevent the compiler adding padding bytes

alignas

- What is the alignas keyword?
  - The alignas keyword was introduced in C++11
  - It forces the compiler to align struct elements on multiples of the word size
- Why is alignas not always helpful when working with binary files?
  - alignas cannot be used for an alignment which is less than the word size
  - It does not prevent the compiler from adding padding bytes