Needs a lot of work space for sorting and decoding.
Sorting of suffixes is tricky to avoid bad cases such as a big file concatenated with itself.
But it typically compresses 20% better than LZW for larger text files.

1999 p6q5 p13q6 MR

\newpage

3 Describe, in detail, how the heap sort algorithm works. [10]

Show that the worst case cost of heapsort is $O(n log n)$. [6]

Would it be possible to implement a variant of heapsort based on a perfectly balanced ternary structure in which the children of node i are at positions $3i-1$, $3i$, and $3i+1$, and if so what would be the advantages and disadvantages of the new method. [4]

Comparative Programming Languages

Answer:
Bookwork -- needs a description of heapify and downheap, and the overall mechanism.
Need to show the initial heapify is O(n) and the final phase is O(n log n).

Last part is yes. More comparisons at each level but the depth of the tree is reduced. Needs division by 3. Better used of on cache stores.

\newpage

Comparative programming Lanugaes

1 Discuss what it means for a program written in C to conform to the ANSI Standard and for a compiler to conform to the standard. [4]

Explore and discuss the reasons why a program conforming to a standard can fail to yield exactly the same results when run on different conforming compilers. [10]

Discuss why it is sometimes possible to write apparently simple expressions, such as {\tt 9+8/3} in PL/1, that yield unexpected results in languages that have a wide variety of numerical data types. for example the expression. To what extent is it possible to eliminate such problems in future languages. [6]

Answer
A conforming program should compile and yield a result that agrees with the standard specification. That standard may not completely define all features of the language, such as the range of integers, or the order of evaluation of operands. Such lack of specification helps the compiler writer to produce more optimal code but can lead to compiler dependent different results.

A conforming compiler will compile any conforming program in accordance with the standard, but may contain extensions to the language.

The second part could explore integer size, floating point details, order of evaluation as in x=++x + ++x.

Talk about BCD decimal and implict type transfers and the difficulty of having rules that are simple, easy to understand and do not produce the occasional oddity. Still happens in modern languages and probably always will especially if the convenience of implicit type transformations is retained. Does it matter?

\newpage

2 Early programming languages had relatively poor facilities for type checking, data abstraction, data hiding and encapsulation. Explain the meaning of these terms and discuss their importance. [6]

Outline the key features that a language must have to be called object-oriented and briefly discuss to what extent C++, Java and Smalltalk have them. [7]

Briefly discuss some of the reasons why C++ programs typically run faster than equivalent programs written in Java or Smalltalk. [7]

Answer:
Type checking -- normally done at compile time, finds simple programming errors, often allowed m/s dependent casts to override the checking.
Data abstraction -- definition of new composite data types including both data fields and operations.

Data hiding -- ability to hide access to the internal fields used to represent a new data type from the user of the data type, thus allowing a change of implementation later.

Classes with sub-class definition using inheritance either single (java) multiple (C++) or possible dynamic (Smalltalk). The ability to give a value that is a subtype of the type expected. Method overriding.
Ideas of class and instance fields and methods.

No bound checking, no garbage collection, features that allow method calls to avoid dynamic lookup so direct (rather than indirect) subroutine jumps can be made (or even inline calls).