Valid CRAM *major.minor* version numbers are as follows:

1.0 The original public CRAM release.

2.0 The first CRAM release implemented in both Java and C; tidied up implementation vs specification differences in *1.0*.

2.1 Gained end of file markers; compatible with *2.0*.

3.0 Additional compression methods; header and data checksums; improvements for unsorted data.

7 Container header structure

The file definition is followed by one or more containers with the following header structure where the container content is stored in the 'blocks' field:

Data type	Name	Value
int32	length	the sum of the lengths of all blocks in this container (headers and data); equal to the total byte length of the container minus the byte length of this header structure
itf8	reference sequence id	reference sequence identifier or -1 for unmapped reads -2 for multiple reference sequences. All slices in this container must have a reference sequence id matching this value.
itf8	starting position on the reference	the alignment start position
itf8	alignment span	the length of the alignment
itf8	number of records	number of records in the container
ltf8	record counter	1-based sequential index of records in the file/stream.
ltf8	bases	number of read bases
itf8	number of blocks	the total number of blocks in this container
array<itf8>	landmarks	the locations of slices in this container as byte offsets from the end of this container header, used for random access indexing. For sequence data containers, the landmark count must equal the slice count. Since the block before the first slice is the compression header, landmarks[0] is equal to the byte length of the compression header.
int	crc32	CRC32 hash of the all the preceding bytes in the container.
byte[ ]	blocks	The blocks contained within the container.

In the initial CRAM header container, the reference sequence id, starting position on the reference, and alignment span fields must be ignored when reading. The landmarks array is optional for the CRAM header, but if it exists it should point to block offsets instead of slices, with the first block containing the textual header.

In data containers specifying unmapped reads or multiple reference sequences (i.e. reference sequence id < 0), the starting position on the reference and alignment span fields must be ignored when reading. When writing, it is recommended to set each of these ignored fields to the value 0.

7.1 CRAM header container

The first container in a CRAM file contains a textual header in ~~a single block, optionally gzip compressed. This text header currently matches the SAM header specification. If compressed, only gzip is allowed as compression method for this block. The CRAM header container does not include a compression header block.~~

~~It is recommended to reserve 50% more space in the CRAM header container than is required for the SAM header text by optionally padding the container with a second raw block consisting of all zeroes. This can be used to subsequently expand the header container in place, such as when updating @SQ records, while~~