,

1. **Primitive types characterizing the machine.**We begin with a definition of the basic types that characterize our presumed JVM like machinei.e., byte addressable, two-complement and where words (W) are 32 bits wide,half words (H) are 16 bits, and bytes (B) are 8 bits.

```
#include <cstdint> // Standard header providing width integer types

typedef int8_t B; // A byte, 8 bits
typedef int16_t H; // Half a word, 16 bits; two bytes
typedef int32_t W; // A full word of 32 bits; four bytes, two half words
```

,

2. **S-Expressions and handles.** An S-expression is either a dotted pair(consistuted by two, smaller, S-expressions) or an atom (which is a string ofcharacters. An S-expression is represented by a 16 bits handle (thetype H) encoding an integer h,

$$2^{15} \leq h \leq 2^{15} - 1$$

The handle of a dotted pair is always positive. The handle of thespecial atom NIL is zero; handles of all other atoms are negative. The sign bitof a handle is thereforea boolean: if it set, then the handle is an atom.

```
struct S { // handle of S expression
  H h; // the handle
};
```

A handle makes it possible to access the representation of the actual data: In the caseof an expression.

,

3. **Represenation of atoms.** Also, for the purpose of our implementation werepresent "atoms" (A) as the address of the first byte in sequence of bytes thatend in the null byte, i.e., a byte in which all bits are zero, denoted in C by `'\0'` and mathematically by $\natural$.

`'\0'` $= \natural =$

0	0	0	0	0	0	0	0

```
typedef const B *const A; // Underlining representation of strings as pointers to bytes
```

,

an unlabeled internal node in the binary tree representation of an S−expression.
A pair is stored in a word containing two half words. The car **and** the cdr parts
of the node. We also pairs to make linked lists, where one half word is the
contents of a list item, **and** the other is a pointer to the subsequent item.
*/

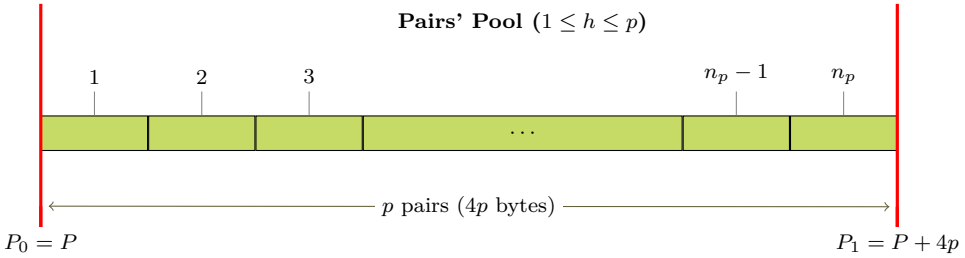
```
union Pair { // The perspectives of a pair.
  W cons: 32; // I. A single word that can
  struct { H car, cdr:16 }; // II. A pair of car and cdr, each in a word.
  struct { H data, next:16}; // III. An item in a linked list
};
```

,

4. Handles as indices. An S-expresion is either a dotted pair (consistuted by two, smaller, S-expressions) or an atom. Such an expression is represented by a half-word that represents an integer h,

$$2^{15} \leq h \leq 2^{15} - 1$$

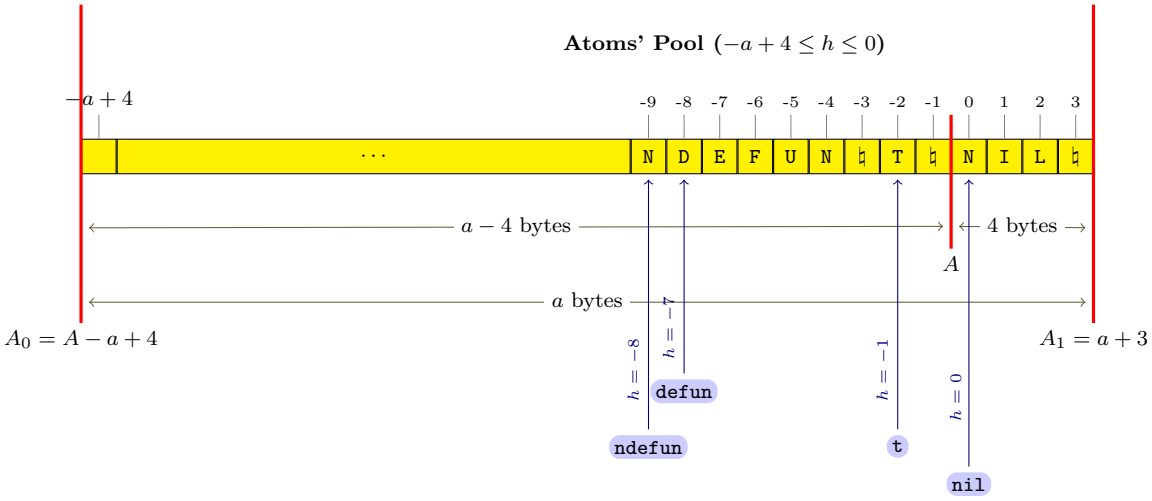
Dotted Pairs If $h > 0$, then the S-experssion that h represents is a dotted-pair. The value of h is interpeted as an index in the pairs' pool: an array P of pairs whose indices are in the range $1, 2, \ldots p$, and where $p > 0$ is the number of pairs in the the statically allocated memory block used for storing pairs.



Atoms If $h \leq 0$, then this S-expression is an atom.The value of case h is interpeted as a, typically negative, index intothe atoms' pool, an array A of a bytes, whose indices are in therange

$$-a + 4, -a + 3, \ldots, 0, \ldots, 3$$

where a is some fixed contant. In other words, A is some fixedmemory address, used to acces a memory block of a bytes that rangesfrom address $A_0 = A - a + 1$ to address $A_1 = A + 3$



,

We need to make array P follow arrary A in memory to conserve as much memory as possible. To **do** so, we allocate a memory block of $a + 4 * p$ bytes, **and** use two perspectives to access it.
*/

Allocate represnetation {
    perspective(byte [a + 4*p])
    perspective(**char** A[a]; pair P[p];)
}

- integers *Positive integers* Positive integer designates a * pair, i.e., an internal node; a negative pair represents a string. A zero * integer designates the NIL atom. There is a bit of trickery to make sure * that if the index zero, the string behind it happens to be NIL. In a sense * the zero is also an index into the strings array. * * Both pools are consecutive in memory: There is a large static buffer in which * both pools reside. */

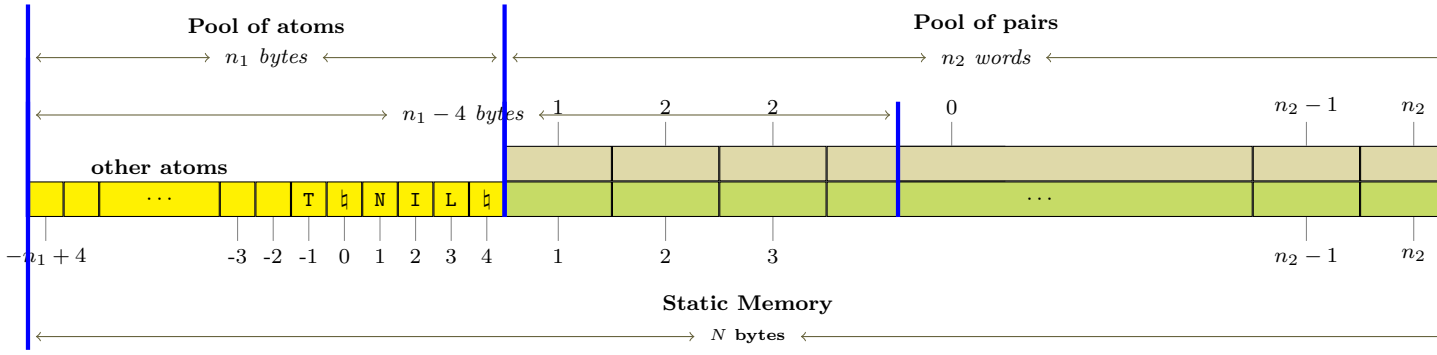
5. Sizes of related to primitive types.

Constants {
$B_n \equiv 8\times$**sizeof** B, *// Eight bits in byte*
$H_n \equiv 8\times$**sizeof** H, *// Sixteen bits in a half word*
$W_n \equiv 8\times$**sizeof** W, *// Thity-two bits in a full word*

$H_c \equiv H_n - 1$, *// Number of bits in the characteristic of type H*
$H_x \equiv 2^{H_c}$*// Maximal positive value representable in type H*
}

6. Static memory. All memory allocations are from a pre-allocated fixedcontigious memory block, the *pool* of n bytes comprising:

1. *Pool of atoms.* A sub-block N_1 *bytes*, from whichinternal representations of atoms are allocated.
2. *Pool of pairs.* A sub-block n_2 *words*,from which internal representations of pairs are allocated.



Constants {
$N_1 \equiv 2^{13}$,
$N_2 \equiv 2^{15}-1$,
$N \equiv N_1\times$**sizeof** B + $N_2\times$**sizeof** W,
}

static B M[N];