CONTIGUOUS ALLOCATION

We would like to simulate what happens in the Memory Management Unit when it handles memory allocation contiguously.

The different data structures are given here as an example but can be modified.

```
#ifndef _MMU_H_
#define _MMU_H_

#define SIZE 65536

typedef short byte_t;
typedef int address_t;

typedef struct hole {
  address_t adr;
  int sz;
  struct hole *next;
  struct hole *prev;
} hole_t;

typedef struct {
  byte_t mem[SIZE];
  hole_t root; // holes list
} mem_t;

// dynamically allocates a mem_t structure and initializes its content
mem_t *initMem();
```

(linked List)

you can use instead a table of flags (easier to implement).

```
// allocates space in bytes (byte_t) using First-Fit, Best-Fit or Worst-Fit
address_t myAlloc(mem_t *mp, int sz);


// release memory that has already been allocated previously
void myFree(mem_t *mp, address_t p, int sz);

// assign a value to a byte
void myWrite(mem_t *mp, address_t p, byte_t val);

// read memory from a byte
byte_t myRead(mem_t *mp, address_t p);

#endif
```

It should be possible to use this memory as indicated in the main :

```
int main() {
  mem_t *mem = initMem();
  address_t adr1 = myAlloc(mem, 5);
  address_t adr2 = myAlloc(mem, 10);
  address_t adr3 = myAlloc(mem, 100);

  myFree(mem, adr2, 10);
  myFree(mem, adr1, 5);

  myWrite(mem, adr3, 543);  // write on the 1st byte
  myWrite(mem, adr3+9, 34); // write on the 10th byte

  byte_t  val1 = myRead(mem, adr3);
  byte_t val2 = myRead(mem, adr3+9);
```

```
}
```

Bonus

Implement the 3 memory allocation algorithms, namely the first-fit, the best-fit and the worst-fit.