Storing Forwarding Data for Moving GCs

When GCs need to move the object, they need to record the new location for the object, at least temporarily

Mark word would encode this for GC code to coordinate the relocation and update-references work.

Garbage Collector는 Object를 이동시켜야 한다.
이때, **"어디로"** 이동 시켜야 하는지에 대한 정보가 **Mark Word에 기록** 되어있다.

Mark Word

객체가 새롭게 이동할 메모리 주소

📌 질문

다른 곳에 메모리 주소 정보를 저장해도 되는가?

📌 답변

그래도 되지만, 설계가 복잡해 진다.
Object의 다른 slot을 사용하게 되면, 부수적인 정보를 더 추가해주어야 한다.

∴참고

You can also bite the bullet and **store the forwarding information completely outside of heap**, like ZGC does.

Storing Object Ages for GCs

```
Fresh object is at d2d6c0f8
*** Move  1, object is at d31104a0
  (object header)  09 00 00 00 (00001001 00000000 00000000 00000000)
                                  ^^^^
*** Move  2, object is at d3398028
  (object header)  11 00 00 00 (00010001 00000000 00000000 00000000)
                                  ^^^^
*** Move  3, object is at d3109688
  (object header)  19 00 00 00 (00011001 00000000 00000000 00000000)
                                  ^^^^
*** Move  4, object is at d43c9250
  (object header)  21 00 00 00 (00100001 00000000 00000000 00000000)
                                  ^^^^
*** Move  5, object is at d41453f0
  (object header)  29 00 00 00 (00101001 00000000 00000000 00000000)
                                  ^^^^
*** Move  6, object is at d6350028
  (object header)  31 00 00 00 (00110001 00000000 00000000 00000000)
                                  ^^^^
*** Move  7, object is at a760b638
  (object header)  31 00 00 00 (00110001 00000000 00000000 00000000)
                                  ^^^^
```

GC에서 살아남게 되면,
AGE값이 하나씩 증가하게 된다.

Identity Hash Code

```
SimpleInt instance = new SimpleInt();
System.out.println(ClassLayout.parseInstance(instance).toPrintable());
```

The HotSpot JVM computes the identity hash code lazily

```
SimpleInt object internals:
 OFFSET  SIZE   TYPE DESCRIPTION                    VALUE
      0     4        (object header)                01 00 00 00 (00000001 00000000 00000000 00000000) (1) # mark
      4     4        (object header)                00 00 00 00 (00000000 00000000 00000000 00000000) (0) # mark
      8     4        (object header)                9b 1b 01 f8 (10011011 00011011 00000001 11111000)
(-134145125) # klass
     12     4    int SimpleInt.state                0
Instance size: 16 bytes
Space losses: 0 bytes internal + 0 bytes external = 0 bytes total
```

```
The identity hash code is 1702146597
SimpleInt object internals:
 OFFSET  SIZE   TYPE DESCRIPTION                    VALUE
      0     4        (object header)                01 25 b2 74 (00000001 00100101 10110010 01110100)
(1957831937)
      4     4        (object header)                65 00 00 00 (01100101 00000000 00000000 00000000) (101)
      8     4        (object header)                9b 1b 01 f8 (10011011 00011011 00000001 11111000)
(-134145125)
     12     4    int SimpleInt.state                0
```

여기는 mark word 영역이다. 즉, Mark Word에 hash code 값 저장