About Dafny Frame and Invariant

Jianhui Chen

2020.5.15

Verification Condition and Invariant

Standard Loop:

```
PRE
while(C)
  // INV
  BODY;
POS
```

Invariant Checking:

```
(Reachable) PRE => INV
(Inductive) INV && C && BODY => INV
(Correct)   INV && !C => POS
```

```
PRE  : the pre-condition of the loop;
C    : the loop condition;
INV  : the loop invariant;
BODY : the loop body;
POS  : the post-condition of the loop;
```

Expression

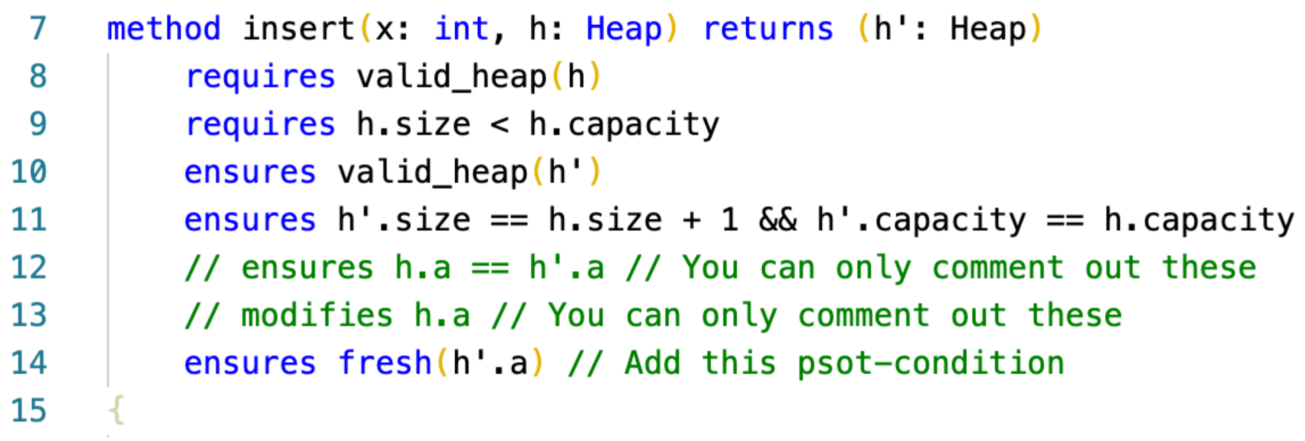
- Old
  - An old expression is used in postconditions.
  - old(e) evaluates to the value expression e had on entry to the current method.
  - *ensure* h.a = *old*(h.a)
- Fresh
  - fresh(e) returns a boolean value that is true if the objects referenced in expression e were all freshly allocated in the current method invocation.
  - The argument of fresh must be either an object reference or a collection of object references.
  - *ensure fresh*(h.a)

Frame

- Reads
  - A reads clause specifies the set of memory locations that a function, lambda, or iterator may read.
  - A method does not have reads clauses because methods are allowed to read any memory.
  - *reads* h.a
- Modifies
  - A modifies clause specifies what memory locations the method or the loop body is allowed to modify.
  - If no modifies clause is given explicitly, there is no memory locations may be modified.
  - *modifies* h.a

Call may violate context's modifies clause M1

Method 1 : New a fresh array.



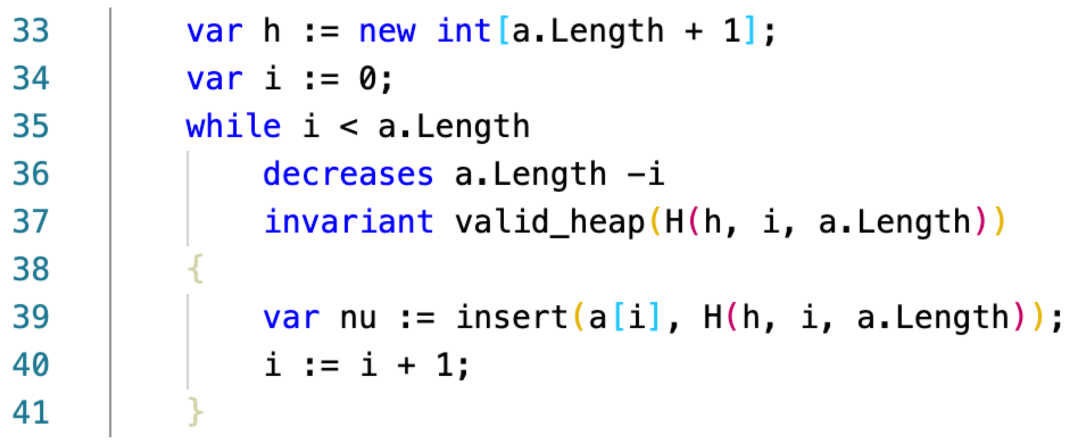
```
method insert(x: int, h: Heap) returns (h': Heap)
    requires valid_heap(h)
    requires h.size < h.capacity
    ensures valid_heap(h')
    ensures h'.size == h.size + 1 && h'.capacity == h.capacity
    // ensures h.a == h'.a // You can only comment out these
    // modifies h.a // You can only comment out these
    ensures fresh(h'.a) // Add this psot-condition
{
```

About Verified Heap – M1

```
    // Copy `h.a` to a fresh array `a`
    var a := new int[h.capacity + 1];
    var i := 1;
    while (i <= h.size)
        decreases h.size - i
        invariant forall j :: 1 <= j < i ==> j <= h.size && a[j] == h.a[j]
    {
        a[i] := h.a[i];
        i := i + 1;
    }
    assert forall i :: 1 <= i <= h.size ==> a[i] == h.a[i];
    // TODO: Fill in the body to satisfy the specification
}
```

Call may violate context's modifies clause M2

Method 2 : Do not use *init_heap* method.



```
var h := new int[a.Length + 1];
var i := 0;
while i < a.Length
    decreases a.Length -i
    invariant valid_heap(H(h, i, a.Length))
{
    var nu := insert(a[i], H(h, i, a.Length));
    i := i + 1;
}
```

Invariant Example

VS Code