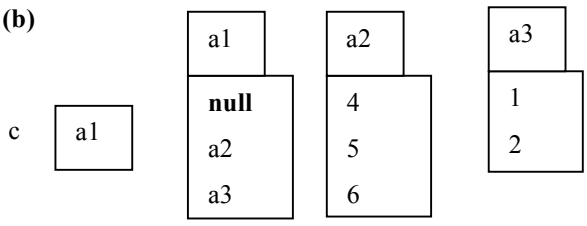
1.
```
x= 0;
for (int k= b.length; 0 != k; k= k – 1) {
    if (b[k–1] == b[k+x–1])
        x= x + 1;
    }
}
```

2.
```
/** = index of the seller in deal[h..k] with best deal.
   Precondition: d[h..k] is not empty*/
private int findBest(int h, int k) {
  int best= h;
  // inv: d[best] is the best deal in d[h..j-1]
  for (int j= h+1; j <= k; j= j+1) {
     Deal n= deal.get(j);
     Deal b= deal.get(h);
     if (n.getPrice() < b.getPrice() ||
         (n.getPrice() == b.getPrice() &&
          n.getCondition() > b.getCondition())) {
       best= j;
     }
  }
  return best;
}
/** Sort the deals for the textbook by price … */
public void sortSellers() {
  // inv: deal[0..k-1] is sorted, and
  //      elements in deal[0..k-1] are better than
  //      elements in d[k..deal.size()-1]
  for (int k= 0; k != deal.size(); k= k+1) {
     int j= findBest(k,deal.size()-1);
     Deal temp= deal.get(j);
     deal.set(j, deal.get(k));
     deal.set(k, temp);
  }
}
```

3.

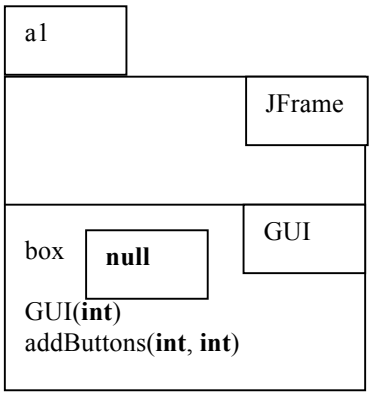
(a) `int[][] b= new int[][] { {1, 3, 6, 10}, {2, 5, 9, 13}, {4, 8, 12, 15}, {7, 11, 14, 16} };`

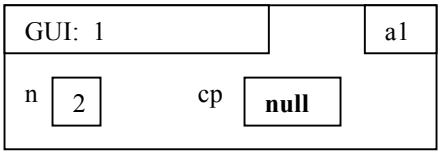
(b)



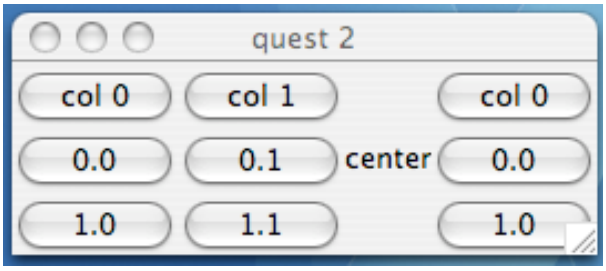
(c) c[2].length

4. (a)





4b.



4c. 9 buttons.

5.
```
/** Constructor: an instance with value k.
    Precondition: k > 0 */
public Positive (int k) {
  this.k= k;
}

/** Constructor: an instance with rational
     number num / denom.
     Precondition: denom != 0 */
public Rational(int num, int denom) {
  super(Math.abs(denom));
  if (denom < 0)
       this.num= – num;
  else  this.num= num
   reduce();
}

/** Set the value of the denominator to n.
  Precondition: n > 0 */
public void setPositive(int n){
  super.setPositive(n);
  reduce();
}
```