1. **import** heapq
2. **import** random
3. **from** collections **import** deque, defaultdict
4. **import** sys
5. **import** copy #use copy.deepcopy to copy data structure!
6. **from** functools **import** cmp\_to\_key
7. input = sys.stdin.readline
8. sys.setrecursionlimit(int(1e6))
10. ##n = int(input())
11. ##a = list(map(int, input().split()))
13. ##自定义排序
14. ##def mycmp(x, y):
15. ##    return -1 if x > y else 1
16. ##
17. ##list.sort(a, key = cmp\_to\_key(mycmp))
18. ##for i in range(0, len(a)):
19. ##    if i > 0:
20. ##        print("",end = ' ')
21. ##    print(a[i], end = '')
22. ##print("")
24. ##alt3注释,alt4取消
25. ##F1看帮助文档
27. ##双端队列写bfs!!
28. ##n, m, sx, sy = map(int, input().strip().split())
29. ##
30. ##dis = [[-1] \* (m + 1) for i in range(n + 1)] ##多维数组酱紫开，深拷贝
31. ##dis[sx][sy] = 0
32. ##que = deque()
33. ##
34. ##dx = [-2, -2, -1, -1, 1, 1, 2, 2]
35. ##dy = [-1, 1, -2, 2, -2, 2, -1, 1]
36. ##
37. ##que.append([sx, sy])
38. ##
39. ##while que:
40. ##    x, y = que.popleft()
41. ##    for d in range(0, 8):
42. ##        nx, ny = x + dx[d], y + dy[d]
43. ##        if nx < 1 or nx > n or ny < 1 or ny > m:
44. ##            continue
45. ##        if dis[nx][ny] == -1:
46. ##            dis[nx][ny] = dis[x][y] + 1
47. ##            que.append([nx, ny])
48. ##
49. ##for i in range(1, n + 1):
50. ##    for j in range(1, m + 1):
51. ##        print(dis[i][j], end = ' ')
52. ##    print("")
54. ##堆的用法& dijkstra最短路
55. ##n, m, s = map(int, input().strip().split())
56. ##adj = [[] for i in range(0, n + 5)]
57. ##for i in range(0, m):
58. ##    x, y, z = map(int, input().strip().split())
59. ##    adj[x].append([y, z])
60. ##
61. ##dis = [(int)(1e18)] \* (n + 1)
62. ##dis[s] = 0
63. ##pq = [[0, s]]
64. ##heapq.heapify(pq)
65. ##while pq:
66. ##    d, nw = heapq.heappop(pq)
67. ##    if d != dis[nw]:
68. ##        continue
69. ##    for v, w in adj[nw]:
70. ##        if dis[v] > d + w:
71. ##            dis[v] = d + w
72. ##            heapq.heappush(pq, [d + w, v])
73. ##for i in range(1, n + 1):
74. ##    print(dis[i], end = ' ')
76. ##从某个库调用方法好一点！码出‘heapq.’后，按tab会出自动补全，good！
77. ##线段树！！
79. n, m = map(int, input().strip().split())
80. a = list(map(int, input().split()))
81. list.insert(a, 0, 0)
83. seg1 = [0] \* (4 \* n + 5)
84. laz1 = [0] \* (4 \* n + 5)
85. seg2 = [0] \* (4 \* n + 5)
86. laz2 = [0] \* (4 \* n + 5)
88. **def** ls(k:int):
89. **return** k << 1
90. **def** rs(k:int):
91. **return** k << 1 | 1
92. **def** push\_up(seg, k:int):
93. seg[k] = seg[ls(k)] + seg[rs(k)]
94. **def** push\_down(seg, laz, l, r, k:int):
95. **if** laz[k]:
96. mid = (l + r) >> 1
97. seg[ls(k)] += laz[k] \* (mid - l + 1)
98. seg[rs(k)] += laz[k] \* (r - mid)
99. laz[ls(k)] += laz[k]
100. laz[rs(k)] += laz[k]
101. laz[k] = 0
102. **def** ask(seg, laz, L, R, l = 1, r = n, k = 1):
103. **if** l > R **or** r < L:
104. **return** 0
105. **if** L <= l **and** r <= R:
106. **return** seg[k]
107. push\_down(seg, laz, l, r, k)
108. mid = (l + r) >> 1
109. **return** ask(seg, laz, L, R, l, mid, ls(k)) + ask(seg, laz, L, R, mid + 1, r, rs(k))
110. **def** upd(seg, laz, L, R, val, l = 1, r = n, k = 1):
111. **if** l > R **or** r < L:
112. **return**
113. **if** L <= l **and** r <= R:
114. seg[k] += val
115. laz[k] += val
116. **return**
117. push\_down(seg, laz, l, r, k)
118. mid = (l + r) >> 1
119. upd(seg, laz, L, R, val, l, mid, ls(k))
120. upd(seg, laz, L, R, val, mid + 1, r, rs(k))
121. push\_up(seg, k)
122. **return**
123. **def** build(seg, laz, l = 1, r = n, k = 1):
124. laz[k] = 0
125. **if** l == r:
126. seg[k]= a[l]
127. **return**
128. mid = (l + r) >> 1
129. build(seg, laz, l, mid, ls(k))
130. build(seg, laz, mid + 1, r, rs(k))
131. push\_up(seg, k)
132. **return**

135. build(seg1, laz1)
136. ##for \_ in range(0, m):
137. ##    opt = list(map(int, input().split()))
138. ##    if opt[0] == 1:
139. ##        upd(seg1, laz1, opt[1], opt[2], opt[3])
140. ##    else:
141. ##        print(ask(seg1, laz1, opt[1], opt[1]))
142. build(seg2, laz2)
143. **for** \_ **in** range(0, m):
144. opt = list(map(int, input().split()))
145. **if** opt[0] == 1:
146. upd(seg2, laz2, opt[1], opt[1], opt[2])
147. **else**:
148. **print**(ask(seg2, laz2, opt[1], opt[2]))