



Tsuen Wan Government Secondary School St. Mark's School
荃灣官立中學 聖馬可中學

2026/27 SDIC Heat Practice 2026/27 SDIC 初賽練習

Time Allowed 限時 2 hours 小時

OVERVIEW 總覽

Section 部分	Format 類型	Total Marks 佔分
A 一	Multiple Choice 單項選擇題	30
B 二	Code Reading 閱讀程序	40
C 三	Complete the Program 完善程序	30
Total 總分		100

REMINDERS 考生注意事項

- The question paper consists of 19 pages and the answer sheet consists of 1 page. The total score is 100 points. Please write your answers on the answer sheet; anything written on the question paper will be considered invalid.
試題共有 19 頁，答題紙共有 1 頁，滿分 100 分。請在答題紙上作答，寫在試題紙上的一律無效。
- No electronic devices (such as calculators, mobile phones, electronic dictionaries, electronic watches, etc.) may be used, nor may any books or reference materials be consulted.
不得使用任何電子設備（如計算機、手機、電子詞典、電子手錶等）或查閱任何書籍資料。

A — Multiple Choice 單項選擇題

Total Marks 佔分

30

- There are 15 questions in total, each worth 2 points, for a total of 30 points
共 15 題，每題 2 分，共計 30 分
- Each question has one and only one correct option
每題有且僅有一個正確選項

1 `g++ {var} brute brute.cpp`

The command above is used to compile the C++ file `brute.cpp` into an executable file named `brute`. In the command, which option should `{var}` be replaced with?

上述指令用於將 C++ 檔案 `brute.cpp` 編譯為名為 `brute` 的執行檔。在該指令中，`{var}` 應該被替換為哪個選項？

- A `-o`
- B `-std`
- C `-O2`
- D `-nm`

2 Consider sorting N numbers, which of the following sorting algorithms has the best asymptotic time complexity in the worst case?

考慮對 N 個數字進行排序，下列哪種排序演算法在最壞情況下具有最佳的漸近時間複雜度？

- A Quick Sort
快速排序
- B Merge Sort
歸併排序
- C Insertion Sort
插入排序
- D Bubble Sort
冒泡排序

- 3 Which of the following sets of operations can delete node `x` from a doubly linked circular list? (Where the `next` field is the node's direct successor and the `prev` field is the node's direct predecessor)
下列哪一組操作能完成從雙向循環鏈表中刪除節點 `x` 的效果？（其中，`next` 域為節點的直接後繼，`prev` 域為節點的直接前驅）
- A `x->prev->next = x->next; x->next->prev = x->prev;`
B `x->prev->prev = x->next; x->next->next = x->prev;`
C `x->next->prev = x->next; x->prev->next = x->prev;`
D `x->prev->prev = x->prev; x->next->next = x->next;`
- 4 The elements $\{7, 3, 6, 8, 2, 1, 4, 5\}$ are inserted sequentially into a binary search tree. Denote the depth of the root as 1, what is the sum of the depths of all nodes?
將元素 $\{7, 3, 6, 8, 2, 1, 4, 5\}$ 依序插入一棵二元搜尋樹。若規定根節點的深度為 1，則所有節點的深度總和是多少？
- A 21
B 22
C 23
D 24
- 5 In the KMP algorithm, for a pattern string `P`, its `next` array `next[i]` is defined as the length of the longest common prefix and suffix of the pattern string `P[1...i]`, and the array index starts from 1. If the length of `P` is 6 and `P[1...3] = abb`, how many distinct `next` arrays are there?
在 KMP 演算法中，對於模式串 `P`，其 `next` 數組 `next[i]` 定義為模式串 `P[1...i]` 最長公共前後綴的長度，且陣列下標從 1 開始。若 `P` 長度為 6，且 `P[1...3] = abb`，則有多少種不同 `next` 數組？
- A 4
B 5
C 12
D 13
- 6 The recursive formula for the time complexity of a certain algorithm is $T(n) = 2T(\sqrt{n}) + O(\log n)$. What is its asymptotic time complexity?
某演算法的時間複雜度的遞歸式為 $T(n) = 2T(\sqrt{n}) + O(\log n)$ ，則其漸進時間複雜度為？
- A $O(\log(n))$
B $O(\log(n) \log(\log(n)))$
C $O(\log^2(n))$
D $O(n)$

- 7 Define $G(A)$ as a weighted complete graph where the edge weight between any pair of vertices (i, j) is $A_i + A_j + |i - j|$. Given that P is a permutation of 1 to 7, what is the difference between the maximum and minimum possible sum of weights in the minimum spanning tree (MST) of all possible $G(P)$?

定義 $G(A)$ 為一個帶權完全圖，其中任意一對頂點 (i, j) 之間的邊權為 $A_i + A_j + |i - j|$ 。已知 P 是 1 到 7 的排列，對於所有可能的 $G(P)$ ，其最小生成樹 (MST) 邊權和的最大可能值與最小可能值之差為？

- A 0
- B 6
- C 9
- D 10

- 8 9 identical candies are distributed among 4 children (some children possibly receive none). The child who receives the most candies must receive at least 3 more candies than the child who receives the fewest. How many different ways are there to distribute candies?

將 9 塊相同的糖果分給 4 個小孩（允許有小孩分不到糖果），要求分到糖果最多的小孩比分到最少的小孩至少多 3 塊，有多少種不同的分配方案？

- A 210
- B 216
- C 256
- D 292

- 9 There are 3 red socks, 4 white socks, and 2 black socks in a drawer. What is the minimum number of socks you need to pull out to guarantee you have two pairs of socks of the same colour (note: the two pairs of socks can be the same colour or different colours)?

抽屜裡有 3 隻紅襪子、4 隻白襪子和 2 隻黑襪子。最少需要抽出多少隻襪子，才能保證湊出兩雙同色的襪子（註：兩雙襪子可以是同一種顏色，也可以是不同顏色）？

- A 4
- B 5
- C 6
- D 7

- 10 For an array initially set to all zeros and of size 10^{12} , which of the following data structures/methods is most suitable to support forced online range addition and single-point query operations?
對於初始全為 0、大小為 10^{12} 的數組，若要支援強制在線的區間加與單點查詢操作，下列哪種資料結構/方法最為適合？
- A Dynamic segment tree
動態開點線段樹
 - B Discretisation
離散化
 - C Block decomposition
分塊
 - D Difference array
差分數組
- 11 What is the maximum number of nodes in a Trie (including the root node) consisting of 10 binary strings of length 5?
由 10 個長度為 5 的 01 字串構成的 Trie 樹（包含根節點），其最大節點數為？
- A 46
 - B 47
 - C 50
 - D 1024
- 12 Which of the following statements about Cartesian trees is incorrect?
關於笛卡爾樹，下列哪一項說法是錯誤的？
- A A Cartesian tree can be constructed using a monotonic stack, with a time complexity of $O(n)$.
使用單調棧能實現笛卡爾樹的構建，其時間複雜度為 $O(n)$ 。
 - B A Cartesian tree satisfies the heap property.
笛卡爾樹滿足堆的性質。
 - C A Cartesian tree is a binary tree.
笛卡爾樹是一棵二元樹。
 - D For a Cartesian tree built from a random permutation of length n , the expected tree depth is on the order of $O(\sqrt{n})$.
由長度為 n 的隨機排列所建構的笛卡爾樹，其期望樹深為 $O(\sqrt{n})$ 。

- 13 How many non-isomorphic, simple, connected bipartite graphs with 6 vertices contain an Eulerian cycle?
具有 6 個頂點的非同構、包含歐拉迴路的二分簡單連通圖共有幾個？
- A 1
B 2
C 3
D 4
- 14 After optimising the recurrence formula using memoization, which of the following formulas experiences the most significant improvement in time complexity? (The formulas apply to positive integers $n > 1$, with $F(0) = F(1) = 1$.)
透過記憶化優化遞推公式後，下列哪一個公式的時間複雜度提升最為顯著？（公式適用於正整數 $n > 1$ ，已知 $F(0) = F(1) = 1$ ）
- A $F(n) = F(\lfloor \frac{n}{2} \rfloor) + F(\lfloor \frac{n}{3} \rfloor)$
B $F(n) = F(n-1) + F(n-2)$
C $F(n) = F(n-1) * n$
D $F(n) = \sum F(2^i)$ when the i -th bit of the binary representation of $n-1$ is 1
 $F(n) = \sum F(2^i)$ 當 $n-1$ 的二進位第 i 位為 1 時
- 15 Let $G(L, R)$ be an undirected graph whose vertices are the integers in the closed interval $[L, R]$, with an undirected edge between two vertices if and only if the two integers are coprime. Find the number of such graphs $G(L, R)$ satisfying $1 \leq L < R \leq 100$ that contain at least one bridge.
定義 $G(L, R)$ 為一個無向圖，其頂點編號為閉區間 $[L, R]$ 內的所有整數，且當且僅當兩個整數互質時，兩點之間存在一條無向邊。求滿足 $1 \leq L < R \leq 100$ 且至少包含一條割邊的 $G(L, R)$ 的數量。
- A 49
B 125
C 148
D 224

B 二 Code Reading 閱讀程序

Total Marks 佔分

40

- Program input will not exceed the defined range of arrays or strings
程序輸入不超過數組或字符串定義的範圍
- For true or false questions, mark ✓ if correct and × if incorrect
判斷題正確填 ✓，錯誤填 ×
- Unless otherwise specified, true or false questions are worth 1.5 points each, multiple choice questions are worth 3 points each, for a total of 40 points
除特殊說明外，判斷題 1.5 分，選擇題 3 分，共計 40 分

(1)

C++

```
1  #include <algorithm>
2  #include <cstdio>
3  using namespace std;
4
5  const int N = 1000;
6  int arr[N], brr[N], sw = 0;
7
8  void selfsort(int l, int r, int id) {
9      if (arr[l] > arr[r]) {
10         swap(arr[l], arr[r]);
11         sw++;
12     }
13     if (id < 0 || r - l + 1 < brr[id]) return;
14     selfsort(l, l + brr[id] - 1, id - 1);
15     selfsort(r - brr[id] + 1, r, id - 1);
16     selfsort(l, l + brr[id] - 1, id - 1);
17 }
18
19 int main() {
20     int n, m;
21     scanf("%d", &n);
22     for (int i = 0; i < n; i++) scanf("%d", arr + i);
23     scanf("%d", &m);
24     for (int i = 0; i < m; i++) scanf("%d", brr + i);
25     sort(brr, brr + m);
26     selfsort(0, n - 1, m - 1);
27     printf("%d\n", sw);
28 }
```

- 16 When *brr* contains 1, runtime error may occur.
當 *brr* 包含 1，可能會出現運行時錯誤。

- 17 When $m \geq \lceil \log_2(n) \rceil$, arr is guaranteed to be ordered when the program terminates
當 $m \geq \lceil \log_2(n) \rceil$ 時，程序終止時，保證 arr 是有序的。
- 18 For any brr that guarantees sorting of array arr , the worst-case time complexity of the algorithm will never be better than bubble sort.
對於所有使程序保證對數組 arr 排序的 brr ，算法的最壞時間複雜度永遠不會優於冒泡排序。
- 19 When the input is 5 4 2 5 1 3 2 2 3, the output result is _____.
當輸入為 5 4 2 5 1 3 2 2 3 時，輸出結果為 _____。
- A 4
B 5
C 6
D 7
- 20 (2.5 marks) Deleting `sort(brr, brr + m);` from line 25, the output will be _____.
(2.5 分) 刪除第 25 行的 `sort(brr, brr + m);`，則輸出會 _____。
- A increased or unchanged
變大或不變
B decreased or unchanged
變小或不變
C absolutely unchanged
一定不變
D indetermined
不確定
- 21 When the input is $n = 6$ and $brr = \{2, 4\}$, the maximum output value is _____.
當輸入的 $n = 6$ 且 $brr = \{2, 4\}$ 時，輸出的最大值為 _____。
- A 3
B 4
C 5
D 6

(2)

C++

```

1  #include <cstdio>
2  #include <vector>
3  #include <queue>
4  using namespace std;
5
6  struct DS {
7      priority_queue<int> pq1;
8      priority_queue<int, vector<int>, greater<int>> pq2;
9      void insert(int x) {
10         if (pq1.empty() || x <= pq1.top()) pq1.push(x);
11         else pq2.push(x);
12         while (pq1.size() > pq2.size() + 1) pq2.push(pq1.top()), pq1.pop();
13         while (pq2.size() > pq1.size()) pq1.push(pq2.top()), pq2.pop();
14     }
15     int query() {
16         return pq1.top();
17     }
18 };
19
20 int dp[105][105], arr[105];
21 int main() {
22     int n, k;
23     scanf("%d %d", &n, &k);
24     for (int i = 1; i <= n; i++) scanf("%d", arr + i);
25     for (int i = 1; i <= n; i++) {
26         fill(dp[i], dp[i] + 105, -1e9);
27         DS st;
28         for (int j = i; j >= 1; j--) {
29             st.insert(arr[j]);
30             int val = st.query();
31             for (int l = 1; l <= k; l++) {
32                 dp[i][l] = max(dp[i][l], dp[j - 1][l - 1] + val);
33             }
34         }
35     }
36     printf("%d\n", dp[n][k]);
37 }

```

- 22 In line 12, changing `pq1.size() > pq2.size() + 1` to `pq1.size() > pq2.size()` will not affect the output of the code.

將第 12 行的 `pq1.size() > pq2.size() + 1` 改為 `pq1.size() > pq2.size()` 後，代碼輸出的結果不會受到影響。

- 23 When the input is $k = 1$, there are at least $\lfloor \frac{n}{2} \rfloor$ elements in the `arr` that are greater than the output.

當輸入的 $k = 1$ 時，`arr` 中至少有 $\lfloor \frac{n}{2} \rfloor$ 個元素大於輸出。

- 24 When the input is $k = n$, the output is the sum of the elements in arr .
當輸入為 $k = n$ 時，輸出為 arr 中元素的總和。
- 25 When the input is 8 2 3 1 4 1 5 9 2 6, the program will output _____.
當輸入為 8 2 3 1 4 1 5 9 2 6 時，程式輸出的答案為 _____。
- A 7
B 9
C 10
D 15
- 26 When the input is 8 3 -3 -1 -4 -1 -5 -9 -2 -6, the program will output _____.
當輸入為 8 3 -3 -1 -4 -1 -5 -9 -2 -6 時，程式輸出的答案是 _____。
- A -1
B -4
C -6
D -10
- 27 If k is $O(n)$, the time complexity of the code is _____.
假設 k 是 $O(n)$ ，代碼的時間複雜度是 _____。
- A $O(n^2)$
B $O(n^2 \log(n))$
C $O(n^3)$
D $O(n^3 \log(n))$

(3)

C++

```
1 #include <cmath>
2 #include <cstdio>
3 #include <numeric>
4 #include <vector>
5 int cnt = 0;
6 int n, k;
7 int check(std::vector<int> L, std::vector<int> R) {
8     cnt++;
9     if (L.size() > R.size()) return -1;
10    if (L.size() < R.size()) return 1;
11    for (int i = 0; i < L.size(); i++) {
12        if(L[i] == abs(k)) return -k / abs(k);
13        if(R[i] == abs(k)) return k / abs(k);
14    }
15    return 0;
16 }
17 bool assert_ans(int ans) {
18     if (ans == k) {
19         printf("You are Right using %d checks\n", cnt);
20         return true;
21     }
22     printf("Wrong answer!\n");
23     return false;
24 }
25 void guess1(int n) {
26     std::vector<int> A(n);
27     iota(A.begin(), A.end(), 1);
28     if (check({A[0]}, {A[1]}) != 0 && check({A[0]}, {A[2]}) != 0) {
29         std::swap(A[0], A[1]);
30     }
31     for (int i = 1; i < A.size(); i++) {
32         int tmp = check({A[0]}, {A[i]});
33         if (tmp != 0) {
34             assert_ans(A[i] * tmp);
35             return;
36         }
37     }
38     assert_ans(0);
39     return;
40 }
41 void guess2(int n) {
42     std::vector<int> A(n);
43     iota(A.begin(), A.end(), 1);
44     if (check({A[0]}, {A.back()}) != 0 && check({A[1]}, {A.back()}) != 0) {
45         std::swap(A[0], A.back());
46     }
47     int b = A.back();
48     A.pop_back();
```

```

49     while (A.size() >= 3) {
50         std::vector<int> G[3], left;
51         while (A.size() % 3) {
52             left.push_back(A.back());
53             A.pop_back();
54         }
55         for (int i = 0; i < A.size(); i++) {
56             G[i % 3].push_back(A[i]);
57         }
58         A = G[2];
59         if (check(G[0], G[1])) {
60             if (check(G[0], G[2])) A = G[0];
61             else A = G[1];
62         }
63         while (left.size()) {
64             A.push_back(left.back());
65             left.pop_back();
66         }
67     }
68     assert_ans(A[0] * check({b}, {A[0]}));
69     return;
70 }
71 int main() {
72     scanf("%d %d", &n, &k);
73     int t;
74     scanf("%d", &t);
75     if (t == 1) guess1(n);
76     else guess2(n);
77     return 0;
78 }

```

(Note: The “number of guesses” below refers to the number of times the `check` function is called (i.e. the value of `cnt`); “correct guess” means the `assert_ans` function `return true` (executes the branch in line 20); all inputs are guaranteed to be $-n \leq k \leq n$ and $3 \leq n \leq 100$.)

(注意：下述的「猜測數」為調用 `check` 函數的次數（即 `cnt` 的值）；「猜測正確」的含義為 `assert_ans` 函數 `return true`（執行第 20 行所在分支）的情況；所有輸入保證 $-n \leq k \leq n$ 且 $3 \leq n \leq 100$ 。）

28 When the input is `6 5 1`, the number of guesses is 5; when the input is `6 5 2`, the number of guesses is 4.
當輸入為 `6 5 1` 時，猜測次數為 5；輸入為 `6 5 2` 時，猜測次數為 4。

29 Regardless of the values of the input n and k , the number of guesses for $t = 2$ is always less than or equal to the number of guesses for $t = 1$.
不管輸入的 n 和 k 具體為多少， $t = 2$ 時的猜測數總是小於等於 $t = 1$ 時的猜測數。

30 Regardless of whether $t = 1$ or $t = 2$, the program will always guess the correct result.
不管 $t = 1$ 或 $t = 2$ ，程式都一定會猜到正確結果。

- 31 When the function `guess1` terminates, the minimum value of `cnt` is _____.
函數 `guess1` 在終止時，`cnt` 的值最少為 _____。
- A 1
B 2
C 3
D 4
- 32 During the execution of the function `guess2`, the order of magnitude of the maximum number of guesses used is _____.
函數 `guess2` 在運行過程中，最多使用的猜測次數的量級為 _____。
- A $O(n)$
B $O(n^2)$
C $O(\sqrt{n})$
D $O(\log n)$
- 33 When the input is $n = 100$, the maximum number of guesses required for $t = 1$ and $t = 2$ in the code are, respectively, _____.
當輸入的 $n = 100$ 的時候，代碼中 $t = 1$ 和 $t = 2$ 分別需要的猜測次數最多分別為 _____。
- A 100, 14
B 100, 13
C 99, 14
D 99, 13

C 三

Complete the Program 完善程序

Total Marks 佔分

30

- Multiple choice questions, 3 points each, for a total of 30 points
單選題，每小題 3 分，共計 30 分

(1) (k -bounded connected pairs) Given a weighted undirected connected graph with n nodes and m edges. A path from u to v is called “strongly k -bounded” if and only if the weights of all edges on the path are at most k , and “weakly k -bounded” if and only if at most one edge on the path has a weight greater than k . Note that a “strongly k -bounded” path is always “weakly k -bounded”. A node pair (u, v) is “strongly k -bounded connected” if and only if there exists at least one “strongly k -bounded” path between u and v . Similarly, it is “weakly k -bounded connected” if and only if there exists at least one “weakly k -bounded” path between u and v . Note that node pair (u, u) is “strongly k -bounded connected” and “weakly k -bounded connected”. Given Q queries, find the number of “strongly k -bounded connected” ordered node pairs and the number of “weakly k -bounded connected” ordered node pairs in the graph for each distinct k . The parameters above satisfy $1 \leq n, m \leq 1000$, $1 \leq q \leq 10$, $1 \leq u, v \leq n$, and $1 \leq w, k \leq 10^9$.

(1) (k -有界連通對) 給定一個含 n 個點、 m 條邊的帶權無向連通圖。對於一條 u 到 v 的路徑，該路徑被稱為「強 k -有界」當且僅當路徑上所有邊的權重至多為 k ，該路徑被稱為「弱 k -有界」當且僅當路徑上至多有一條邊的權重大於 k 。注意，「強 k -有界」路徑總是「弱 k -有界」的。對於點對 (u, v) ，該對被稱為「強 k -有界連通」當且僅當存在至少一條由 u 到 v 的「強 k -有界」路徑；同樣地，該對被稱為「弱 k -有界連通」當且僅當存在至少一條由 u 到 v 的「弱 k -有界」路徑。注意，點對 (u, u) 是「強 k -有界連通」且「弱 k -有界連通」的。給定 Q 個查詢，對於不同的 k ，分別求出圖中「強 k -有界連通」的有序點對的數量和「弱 k -有界連通」的有序點對的數量。上述參數滿足 $1 \leq n, m \leq 1000$, $1 \leq q \leq 10$, $1 \leq u, v \leq n$ 和 $1 \leq w, k \leq 10^5$ 。

The following code solves the above problem. Try to complete the program.

以下代碼求解了上述問題。試補全程序。

C++

```
1 #include <algorithm>
2 #include <array>
3 #include <cstdio>
4 #include <deque>
5 #include <numeric>
6 #include <vector>
7 int p[2005], W[2005], cnt[100005];
8 bool vis[1005];
9 std::vector<std::pair<int, int>> G[1005];
10 std::vector<int> T[2005];
11 int find(int u) {
12     return p[u] == u ? _____ 1 _____;
13 }
14 int dfs(int u) {
15     if (T[u].empty()) return 1;
16     int a = dfs(T[u][0]), b = dfs(T[u][1]);
17     cnt[W[u]] += _____ 2 _____;
18     return a + b;
19 }
```

```
20 int main() {
21     int n, m, q;
22     scanf("%d %d %d", &n, &m, &q);
23     std::iota(p, p + 2005, 0);
24     std::vector<std::array<int, 3>> eds;
25     for (int i = 0; i < m; i++) {
26         int u, v, w;
27         scanf("%d %d %d", &u, &v, &w);
28         G[u].emplace_back(v, w);
29         G[v].emplace_back(u, w);
30         eds.push_back({w, u, v});
31     }
32     sort(eds.begin(), eds.end());
33     int ttl = n;
34     for (auto [w, u, v] : eds) {
35         u = find(u); v = find(v);
36         if (u == v) continue;
37         T[++ttl] = {u, v};
38         p[u] = p[v] = ttl;
39         W[ttl] = w;
40     }
41     dfs(ttl);
42     for (int i = 1; i <= 100000; i++) cnt[i] = _____ 3 _____;
43     while (q--) {
44         int k;
45         scanf("%d", &k);
46         int ans = 0;
47         for (int i = 1; i <= n; i++) {
48             std::fill(vis, vis + n + 1, 0);
49             std::deque<std::pair<int, int>> dq;
50             dq.emplace_back(i, 0);
51             while (!dq.empty()) {
52                 auto [u, d] = dq.front(); dq.pop_front();
53                 if (_____ 4 _____) continue;
54                 vis[u] = true;
55                 ans++;
56                 for (auto [v, w] : G[u]) {
57                     if (w <= k) dq.emplace_front(v, d);
58                     else dq.emplace_back(_____ 5 _____);
59                 }
60             }
61         }
62         printf("%d %d\n", cnt[k] * 2 + n, ans);
63     }
64 }
```

34 $\frac{1}{1}$ should be filled with
處應填

- A `u : p[u] = find(u)`
- B `u : p[u] = find(p[u])`
- C `p[u] = find(u) : u`
- D `p[u] = find(p[u]) : u`

35 $\frac{2}{2}$ should be filled with
處應填

- A `(a - 1) * (b - 1)`
- B `a * b - a - b`
- C `a * b - 1`
- D `a * b`

36 $\frac{3}{3}$ should be filled with
處應填

- A `cnt[i - 1] + cnt[i]`
- B `cnt[i - 1] * cnt[i]`
- C `cnt[i] * cnt[i]`
- D `cnt[i] * (cnt[i] + 1) / 2`

37 $\frac{4}{4}$ should be filled with
處應填

- A `vis[u] || d > 1`
- B `vis[u] || d > k`
- C `vis[u] && d > 1`
- D `vis[u] && d > k`

38 $\frac{5}{5}$ should be filled with
處應填

- A `v, w`
- B `v, d + 1`
- C `v, d + w`
- D `v, w / k`

(2) (Minimum Coprime Cost) Given an array arr of length n . Define an array A of length $n \times m$ and $A_i = arr_{i \bmod n}$ (indices start from 0). Let B be an array of length $n \times m$ such that $B_i \geq 0$ and $|A_i - B_i| \leq 40$, and for all $i > 0$, B_{i-1} and B_i are coprime (i.e., $\gcd(B_{i-1}, B_i) = 1$). If the cost is defined as $\sum |A_i - B_i|$, find the minimum cost. The parameters above satisfy $1 \leq n \leq 100$ and $1 \leq m, arr_i \leq 10^9$. It is guaranteed that at least one valid B exists.

(2) (最小互質代價) 給定一個長度為 n 的數組 arr 。定義數組 A ，其長度為 $n \times m$ ，且 $A_i = arr_{i \bmod n}$ （下標從 0 開始）。設 B 為長度為 $n \times m$ 的數組，滿足 $B_i \geq 0$ 和 $|A_i - B_i| \leq 40$ ，且對所有 $i > 0$ ， B_{i-1} 與 B_i 互質（即 $\gcd(B_{i-1}, B_i) = 1$ ）。若代價定義為 $\sum |A_i - B_i|$ ，求最小代價。上述參數滿足 $1 \leq n \leq 100$ 和 $1 \leq m, arr_i \leq 10^9$ 。保證至少存在一個合法的 B 。

The following code solves the above problem. Try to complete the program.

以下代碼求解了上述問題。試補全程序。

C++

```

1  #include <algorithm>
2  #include <cstdio>
3  #include <numeric>
4  #include <vector>
5  const int RNG = 40, R = 2 * RNG + 1;
6  const long long INF = 1e18;
7  int n, m;
8  int arr[105];
9  inline bool check(int idx, int j, int k) {
10     int v1 = arr[idx] + j - RNG;
11     int v2 = arr[idx + 1] + k - RNG;
12     return (v1 >= 0 && v2 >= 0 && std::gcd(v1, v2) == 1);
13 }
14 struct Matrix {
15     long long mat[R][R];
16     Matrix() {
17         for (int i = 0; i < R; ++i)
18             for (int j = 0; j < R; ++j)
19                 mat[i][j] = INF;
20     }
21     static Matrix mul(const Matrix& A, const Matrix& B) {
22         Matrix C;
23         for (int k = 0; k < R; ++k)
24             for (int i = 0; i < R; ++i)
25                 for (int j = 0; j < R; ++j)
26                     C.mat[i][j] = _____;
27         return C;
28     }
29 };
30 Matrix qpow(Matrix A, long long p) {
31     Matrix res = A;
32     p--;
33     while (p > 0) {
34         if (p & 1) res = Matrix::mul(res, A);
35         A = _____;
36         p >>= 1;

```

```
37     }
38     return res;
39 }
40 std::vector<long long> run_dp(int begin_val) {
41     std::vector<long long> dp(R, INF);
42     for (int k = 0; k < R; ++k) {
43         if (begin_val == -1 || check(n - 1, begin_val, k))
44             dp[k] = abs(k - RNG);
45     }
46     for (int j = 1; j < n; ++j) {
47         std::vector<long long> ndp(R, INF);
48         for (int k = 0; k < R; ++k)
49             for (int l = 0; l < R; ++l)
50                 if (____ 3 ____ )
51                     ndp[k] = std::min(ndp[k], ____ 4 ____ );
52         dp = move(ndp);
53     }
54     return dp;
55 }
56 int main() {
57     scanf("%d %d", &n, &m);
58     for (int i = 0; i < n; ++i) scanf("%d", arr + i);
59     arr[n] = arr[0];
60     std::vector<long long> vec = run_dp(-1);
61     if (m == 1) {
62         printf("%lld\n", *min_element(vec.begin(), vec.end()));
63         return 0;
64     }
65     Matrix T;
66     for (int i = 0; i < R; ++i) {
67         if (arr[n - 1] + i - RNG < 0) continue;
68         std::vector<long long> dp_i = run_dp(i);
69         for (int j = 0; j < R; ++j) T.mat[j][i] = dp_i[j];
70     }
71     Matrix res = ____ 5 ____ ;
72     long long ans = INF;
73     for (int i = 0; i < R; ++i)
74         for (int j = 0; j < R; ++j)
75             ans = std::min(ans, vec[i] + res.mat[j][i]);
76     printf("%lld\n", ans);
77     return 0;
78 }
```

39 $\frac{1}{1}$ should be filled with
處應填

- A `C.mat[i][j] + std::min(A.mat[i][k], B.mat[k][j])`
- B `C.mat[i][j] + A.mat[i][k] * B.mat[k][j]`
- C `std::min(C.mat[i][j], A.mat[i][k] + B.mat[k][j])`
- D `std::min(C.mat[i][j], A.mat[i][k] * B.mat[k][j])`

40 $\frac{2}{2}$ should be filled with
處應填

- A `Matrix::mul(A, 2)`
- B `Matrix::mul(A, A)`
- C `Matrix::mul(res, 2)`
- D `Matrix::mul(res, res)`

41 $\frac{3}{3}$ should be filled with
處應填

- A `check(j - 1, k, 1)`
- B `check(j - 1, 1, k)`
- C `check(j, k, 1)`
- D `check(j, 1, k)`

42 $\frac{4}{4}$ should be filled with
處應填

- A `dp[k] + abs(k - RNG)`
- B `dp[k] + abs(1 - RNG)`
- C `dp[1] + abs(k - RNG)`
- D `dp[1] + abs(1 - RNG)`

43 $\frac{5}{5}$ should be filled with
處應填

- A `qpow(T, m - 1)`
- B `qpow(T, m)`
- C `qpow(T, (long long)n * m - 1)`
- D `qpow(T, (long long)n * m)`