

Krempita

For his algorithms homework, Faruk was given the task of writing a program that sorts an array of zeroes and ones of arbitrary length. We call such an array of zeroes and ones a **binary string** of length n , and we call the characters at positions $0, 1, \dots, n-1$ the **bits** of the string. A binary string is sorted if all zeroes appear before all ones.

Unfortunately, Faruk did not pay attention during the lecture because he was thinking about krempita, so he does not know the concrete string that needs to be sorted. Therefore, he wants a program that works correctly for every possible binary string of length n , and consists only of a sequence of **conditional swaps**. A conditional swap (A_i, B_i) swaps the bits at indices A_i and B_i only if $S[A_i] > S[B_i]$; otherwise, it does nothing. The swaps are performed in order, from the first to the last.

Faruk only knows the length of the string, n . The task is to find a sequence of conditional swaps $(A_1, B_1), (A_2, B_2), \dots, (A_M, B_M)$, where $0 \leq A_i, B_i \leq n-1$, such that the string S is sorted after all swaps are performed, regardless of the initial contents of S .

For example, for $n = 2$, the sequence consisting of one swap $\{(0, 1)\}$ is enough. If $S = 10$, then $S[0] = 1 > 0 = S[1]$, so the bits are swapped and we get $S = 01$. For all other strings (00, 01, 11), the swap is not performed, and they are already sorted.

Help Faruk construct a sequence with as few swaps as possible.

Input

The first line contains the number of test cases T ($1 \leq T \leq 99$).

Each of the next T lines contains one integer n_i ($2 \leq n_i \leq 100$), the length of the binary string that needs to be sorted.

Output

For each test case, first print the number of swaps M_i . Then print M_i lines, each containing two integers A_i and B_i describing the i -th conditional swap. It must hold that $M_i \leq 10^4$.

Scoring

There are 4 test cases. The first one is the same as the sample below and is not scored.

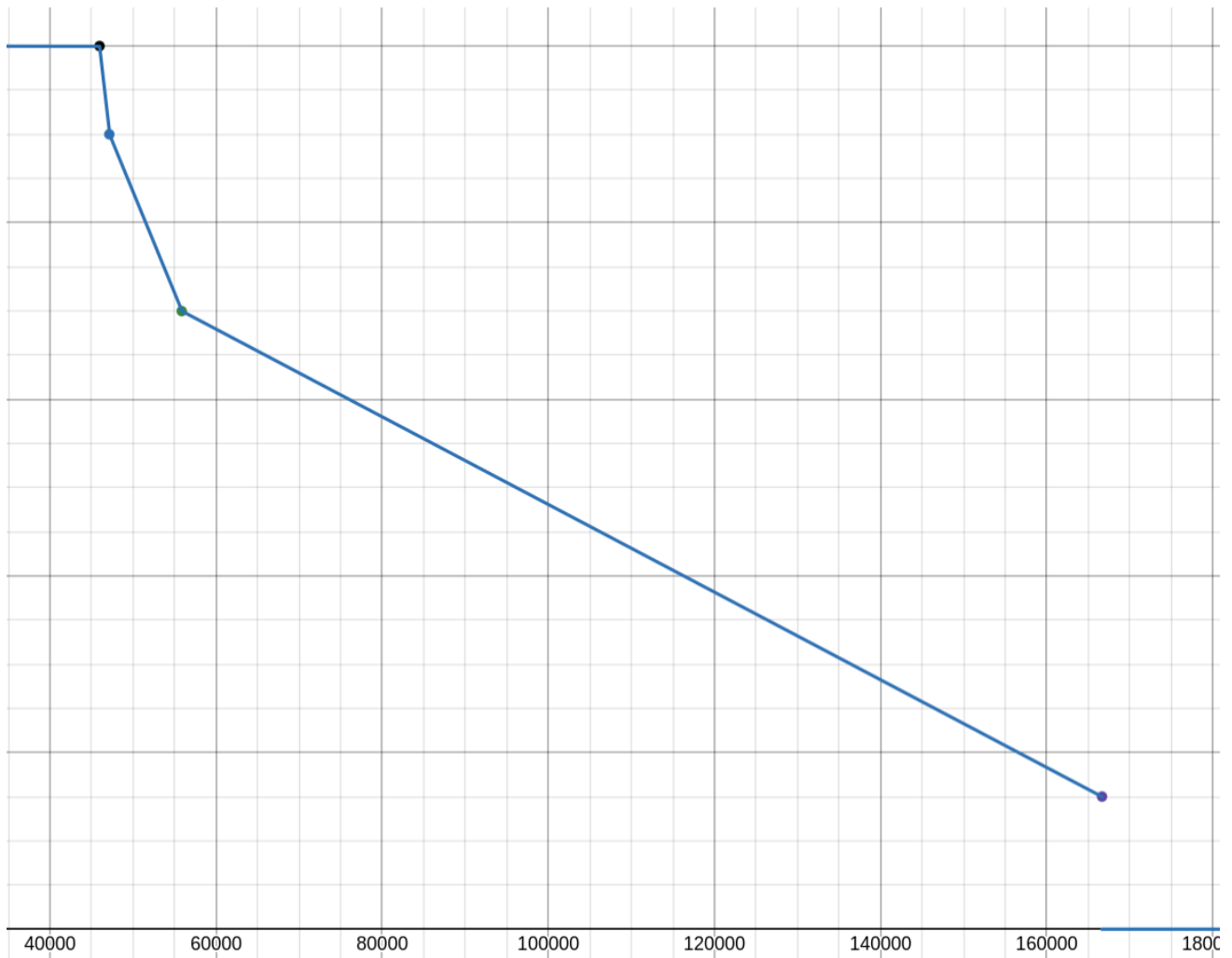
The second, third and fourth test cases are the only scored ones. Let $M = M_1 + \dots + M_T$ be the total number of swaps used for a particular test case.

Test case number	Test case description	Scoring description
2	$T = 4, n_i = i + 1$	$10 \cdot \max(0, (22 - M))/4$ points
3	$T = 99, n_i = i + 1$; for every string S there exists an index k such that $S[i] \leq S[k]$ for $i < k$ and $S[i] \geq S[k]$ for $i > k$	$20 \cdot f(M)$ points
4	$T = 99, n_i = i + 1$	$70 \cdot f(M)$ points

where $f(M)$ is defined as:

$$f(M) = \begin{cases} 1, & M < 45954 \\ 1 - \frac{0.1}{1197}(M - 45954), & 45954 \leq M \leq 47151 \\ 0.9 - \frac{0.2}{8703}(M - 47151), & 47151 < M \leq 55854 \\ 0.7 - \frac{0.55}{110796}(M - 55854), & 55854 < M \leq 166650 \\ 0, & M > 166650 \end{cases}$$

Graph of the function $f(M)$:



Examples

Input 1

```
2
2
3
```

Output 1

```
1
0 1
3
0 2
0 1
1 2
```

Explanation 1

For a string of length 2, one swap $(0, 1)$ is enough, as described in the statement.

For a string of length 3: if $S = 010$, after the swap $(0, 2)$ the string remains 010 because $S[0] = 0 = S[2]$, so the condition is not satisfied. After the swap $(0, 1)$, the string remains 010 because $S[0] = 0 < 1 = S[1]$, so the condition is not satisfied. After the swap $(1, 2)$, the bits at indices 1 and 2 are swapped because $S[1] = 1 > 0 = S[2]$, and we get $S = 001$.