TWGSS OI Team

2024/25 Team Selection Test

Editorial

Wong Sheung Chit Lo Ting Wai Wong Ho Ki

7 May 2025

Statistics

	Problem	Attempts	Full	Max	Mean	Std Dev	LQ	Median	UQ	Subtasks	First Solve	
TS250	Top Scorer	33	10	10	8.636	3.32	10	10	10	5 | 29	2 | 28	3 | 28	Fung Hiu Nok	00:00:54
TS251	Treasure Passage	28	30	30	17.214	13.061	0	22	30	5 | 20	7 | 18	10 | 16	8 | 12	Hui Ka Wai	00:06:52
TS252	Timing Challenge	18	35	35	23.5	16.276	0	35	35	3 | 13	7 | 12	12 | 12	13 | 12	Hui Ka Wai	00:13:25
TS253	Test Case	21	45	45	29.524	20.639	0	45	45	5 | 15	10 | 14	15 | 14	7 | 13	5 | 13	3 | 13	Fung Hiu Nok	00:01:49
TS254	Tutor Assignment	20	50	50	23.65	24.148	0	11.5	50	5 | 11	8 | 9	10 | 9	13 | 10	14 | 9	Fung Hiu Nok	00:04:34
TS255	Typical PTCG	22	50	50	31	23.423	0	50	50	6 | 15	8 | 14	12 | 14	13 | 13	11 | 13	Nip Kit Fung	00:16:51
TS256	Tick-tock Light	12	65	27	8.833	12.502	0	0	26	12 | 1	13 | 4	14 | 3	15 | 0	11 | 0		
TS257	Trivial Colour	6	80	80	45.333	36.271	0	56	80	18 | 3	20 | 3	17 | 4	15 | 4	10 | 3	Fung Hiu Nok	00:15:08
TS258	Travelling Wonder	7	135	55	10.714	18.599	0	0	10	10 | 3	21 | 1	24 | 1	19 | 0	16 | 0	22 | 0	23 | 0		
		33	**500**	**374**	**103.576**	**116.171**	**10**	**30**	**204**			

Main Characters

Alice

Bob

Chris Wong
(You)

Top Scorer

TS250

Problem Statement

Given Alice's score A Bob's score B

Output the person's name with higher score.
If both scores are the same, output `Draw` instead.

Note that A and B are given in 2 decimal places, even if it is an integer.

Subtask 1

$B = 0.00$

There are only 2 situations:

1. $A = 0$ → `Draw`
2. $A > 0$ → `Alice`

How to implement?

You should know if you are sitting here, except *some people*...

Ok, use `float` / `double` and an if-else statement. I can't help you if you still don't know how.

Expected Score	5	Cumulative	5

Full Solution

We have 3 cases:

1. $A > B$ → `Alice`
2. $A < B$ → `Bob`
3. $A = B$ → `Draw`

This is actually a tuned down version of W008 Pick a Card, just have some simple if-else statements and you've AC the problem!

Expected Score	**10**	Cumulative	10

Subtask 2

$0.00 \leq A, B \leq 9.99$

A safety net for you in case you somehow used `string` instead of floating-point number.

Why `string` works for this case?

Because the format of A and B are fixed to `a.aa` and `b.bb` respectively.

As `string` used dictionary-ordered comparison, it is clear that the compare result will be the same as that of the number comparison result.

Expected Score	2

Common Mistakes

```cpp
#include <bits/stdc++.h>
using namespace std;

int main(){
  int A, B;
    cin >> A >> B;
    if (A == B){
        cout << "Draw";
    }else if (A < B){
        cout << "Bob";
    }else{
        cout << "Alice";
    }
}
```

We are not comparing integers!

```cpp
#include <bits/stdc++.h>
using namespace std;

int main(){
    float a, b;
  cin >> a >> b;
    if (a>b){
        cout << "Alice";
    }else if (a<b){
        cout << "Bob";
    }else if (a=b){
        cout << "Draw";
    }
}
```

Please refer to
2024/25 C++ Foundation Course slides
Chapter 3 Page 6

Treasure Passage

TS251

Problem Statement

There are N chests,
the i^{th} chest has the coordinate (X_i, Y_i).

Find the index of the closest chest to $(0, 0)$
and the corresponding distance.

Manhattan distance

Sum of difference between the x- and y-coordinate of two points

Subtask 1

$N = 1$
$0 \leq X_i, Y_i \leq 5000$

In this case, there is only 1 chest.
Therefore, the index of the closest chest must be 1.

As $X_i, Y_i \geq 0$, the distance between the chest can be evaluated by $X_1 - 0 + Y_1 - 0 = X_1 + Y_1$.

You only need to output it with the correct output format!

O(1)	Expected Score	5	Cumulative	5

Subtask 2

$N = 2$
$0 \leq X_i, Y_i \leq 5000$

Now, there are 2 chests.

So we need to compare which chest is closer to $(0, 0)$.

How can we find the distance between each chest and $(0, 0)$?

Right! It is just what Subtask 1 is doing!

But which chest should I choose?

Subtask 2

After we have calculated the distance between each chest and $(0, 0)$,
we can compare it and choose the smaller one!

If the distance of the 1^{st} chest is less than that of the 2^{nd} chest,
we just need to output an index of `1`,
otherwise output an index of `2`.

Finally output the corresponding distance.

O(1)	Expected Score 7	Cumulative 12

Subtask 3

$1 \leq N \leq 2000$
$0 \leq X_i \leq 5000$
$Y_i = 0$

Wow! In this case, there are at most 2000 chests!

We can't simply use a single if statement to solve it!

We should use for loop to solve it!

But how can we find the minimum distance among the chests?

Subtask 3

We can make two integer variables for keeping track of the index and the shortest distance respectively!

We can initialise the value of the index variable be -1 and the shortest distance be 5001 (as the longest distance exists is 5000).

Subtask 3

Using for loop, each time we input the x-coordinate of the chest, we can simply compare the x-coordinate of the chest and the shortest distance (as the distance between the chest and $(0, 0) = X_i + 0 = X_i$).

Each time when the x-coordinate of the chest $<$ the variable of the shortest distance, we can change the shortest distance to the x-coordinate and update the value of the variable of the index into the current one.

Using this method, we can compare all the distance between the chests and (0, 0). Finally, output the index and the shortest distance!

O(N)	Expected Score	**10**	Cumulative	22

Full Solution

Recall how we can find the Manhattan distance, it is simple calculated by adding the difference between the x-coordinate and the y-coordinate of two points.

As one of the point is $(0, 0)$ in this question, the formula becomes:

$$|X_i - 0| + |Y_i - 0| = |X_i| + |Y_i|$$

How can we find the absolute value of a number? There are several methods, where the easiest method is using C++ built in `abs()` function.

Therefore, the Manhattan distance between the chest with coordinates (X_i, Y_i) and $(0, 0)$ can be written as `abs(X_i)+abs(Y_i)` in code.

Full Solution

Then how can we find the shortest distance?

Using the method in Subtask 3!

We can make two variables to save the index and the shortest distance.

Once the distance of the current chest $<$ the shortest distance, update both the value into the current one.

Finally we can get the index and the corresponding distance.

O(	N	)	Expected Score	**30**	Cumulative	30

Timing Challenge

TS252

Problem Statement

There is a stopwatch that the clock hand takes K seconds for it to travel a full cycle.

There are N rounds in the timing challenge.

In each round, the clock hand points exactly upwards initially. Then the clock hand will move anti-clockwise for A_i seconds and move clockwise for C_i seconds.

If the clock hand ends up pointing exactly upwards, you would gain 1 point. You are asked to find the point you get into the whole challenge.

Subtask 1

$N = 1$
$0 \leq A_i, C_i < K$

In this case, we can simply count for looping $A_i + C_i$ times to find the where the clock hand points exactly.

We can assume clockwise is positive and anti-clockwise is negative.

Then each time we can add by 1 from looping the 1^{st} to the ${A_i}^{\text{th}}$ time, and reduce by 1 from looping the $(A_i + 1)^{\text{th}}$ to the $(A_i + C_i)^{\text{th}}$ time.

Subtask 1

Then when will the clock hand point upwards?

When the final value is equal to 0, which means the clock hand hasn't moved.

O(K)	Expected Score	3	Cumulative	3

Observation 1

When will the value become 0?

We can find it from the equation!

Assume $0 \leq A_i, C_i < K$,

$$A_i(1) + C_i(-1) = 0$$
$$A_i - C_i = 0$$
$$A_i = C_i$$

So, only when $A_i = C_i$,the value will become 0.

Subtask 2

$A_1 = A_2 = \cdots = A_N$
$0 \leq A_i, C_i < K$

From the observation, we can see that only one value of C_i will satisfy the equation.

Therefore, we just need to count the number of i such that $C_i = A_\mathrm{i} = A_2 = \cdots = A_N$.

O(N)	Expected Score	**10**	Cumulative	10

Subtask 3

$0 \leq A_i, C_i < K$

It seems like Subtask 2. However, the values of A_i are not equal.

Therefore, we need to check those values one by one.

The only thing we need to do is counting the number of $A_i = C_i$ for each i.

O(N)	Expected Score	**22**	Cumulative	22

Observation 2

Assume the stopwatch takes 4 seconds for it to travel a full cycle.

Is there any difference between moving anti-clockwise for 1 second and for 5 seconds? There are finally pointing to the same position!

Therefore, for a stopwatch that taking K seconds for it to travel a full cycle, the position of

moving anti-clockwise for A_i seconds

$=$

moving anti-clockwise for $A_i \% K$ seconds.

Full Solution

To easily solve this problem, we can divide the problem into two parts.

The first one is converting the value of A_i and C_i such that $0 \leq A_i, C_i < K$.

The second one is to check if the clock hand points upwards exactly.

To do the first part, we can apply Observation 2!

We can change each of the value of A_i and C_i become $A_i \% K$ and $C_i \% K$ respectively, which satisfy $0 \leq A_i, C_i < K$.

How can we do the second part? It just likes doing Subtask 3!

Check if $A_i = C_i$. If yes, then count it!

O(	N	)	Expected Score	**35**	Cumulative	35

Test Case

TS253

Problem Statement

Change the given string S to title case,

i.e. for each word in S,

> the first letter need to be in uppercase (`A`-`Z`), and
> the rest of the letters are in lowercase (`a`-`z`).

Each word in S is separated by a space.

Examples

`TeST case`	→	`Test Case`
`NEver gonNA RaIn elDEr`	→	`Never Gonna Rain Elder`

Key Ideas

`TeST case` → `Test Case`

`NEver gonNA RaIn elDEr` → `Never Gonna Rain Elder`

We immediately have some questions to solve:

1. How can we input such a string S,
   where each word in it is separated by a space?
2. How can we detect if the character is the first letter of a word?
3. How can we modify the letter cases correctly?

Let's use the `NEver gonNA RaIn elDEr` sample
to tackle these questions one by one.

KEY IDEA 1

Input with Spaces

1. How can we input such a string S,
   where each word in it is separated by a space?

We know that using `cin >> S`
will just input the string `S` up to the first space.

Input	`NEver gonNA RaIn elDEr`
`S` when using `cin >> S`	`NEver`

How can we do input with spaces?

Using the `getline()` function!

KEY IDEA 1

Input with Spaces

1. How can we input such a string S,
   where each word in it is separated by a space?

As you should have learnt before...
(2024/25 C++ Foundation Course slides Chapter 10 Page 14-19)

Using `getline(cin, S)` will input a whole line of string to `S`,

which is exactly what we need in this problem.

Input	`NEver gonNA RaIn elDEr`
`S` when using `cin >> S`	`NEver`
`S` when using `getline(cin, S)`	`NEver gonNA RaIn elDEr`

KEY IDEA 2

Identify First Letter

2. How can we detect if the character is the first letter of a word?

After using `getline(cin, S)` to input the string,
now we have `S` equals to `NEver gonNA RaIn elDEr`.

Note that before every first letter of a word, there is a space before it.
(Except for the first word, i.e. `NEver`)

We can make great use of this property to identify those first letters!

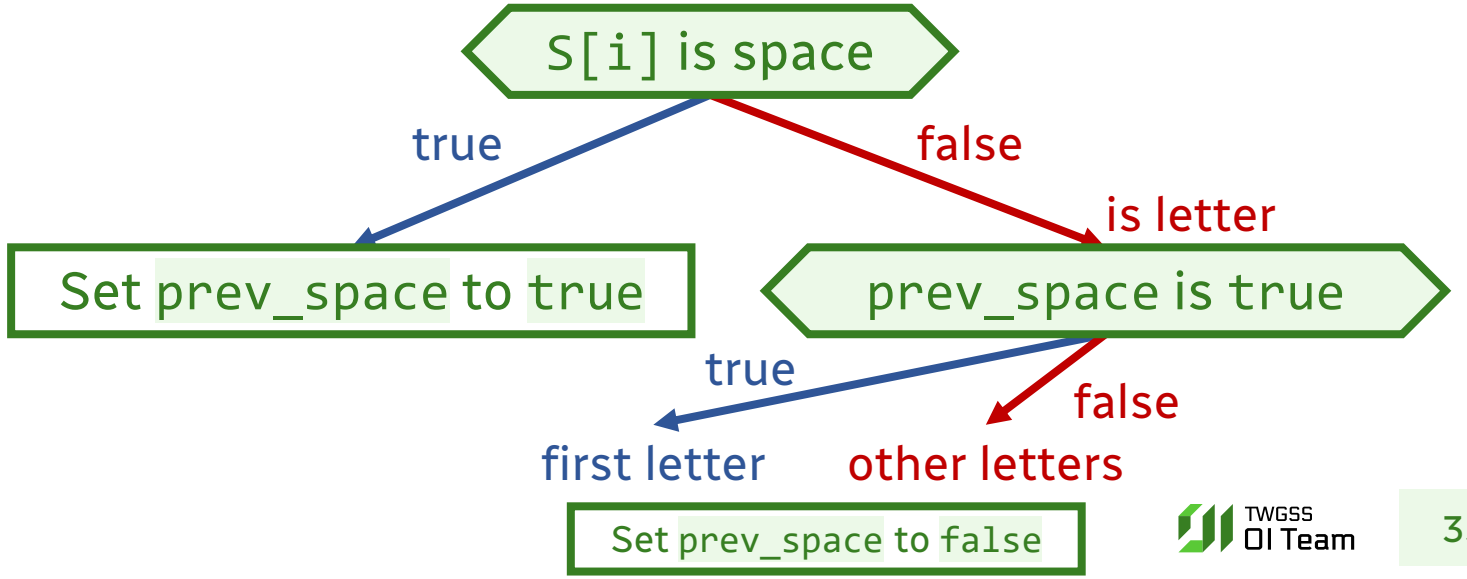
```cpp
bool prev_space = true;
for (int i=0; i<S.size(); i++){
  if (S[i] == ' '){
    // space
    prev_space = true;
  }else{
    // letters
    if (prev_space){
      // first letter of a word
    }else{
      // other letters of a word
    }
    prev_space = false;
  }
}
```

KEY IDEA 2

Identify First Letter

Here, we use a `bool` variable named `prev_space` to store if the previous character is a space (or if it's the first character of `S`).

Then we use a for loop to loop through each character. For the i^{th} character `S[i]`:



KEY IDEA 1+2

Input with Spaces + Identify First Letter

1. How can we input such a string S,
   where each word in it is separated by a space?
2. How can we detect if the character is the first letter of a word?

The method combining `getline()` and a `bool` variable works perfectly!

However, is there a simpler method?

Of course!

May I introduce our old friend `cin`, with its great partner while loop!

(2024/25 C++ Foundation Course slides Chapter 10 Page 20-22)

KEY IDEA 1+2

Input with Spaces + Identify First Letter

```cpp
string S;
while (cin >> S){
  // S stores one word
  for (int i=0; i<S.size(); i++){
    if (i == 0){
      // first letter of the word
    }else{
      // other letters of the word
    }
  }
  cout << S << " ";
}
```

We can use `while (cin >> S)` to input one word at a time. For each iteration of the while loop, `S` stores one word.

We loop through the letters in each word using a for loop, where when `i` equals to `0` means it is the first letter.

Note that each word is separated by **one space**. Thus after handling each word separately, we can add one space between each processed word in output without problem.

KEY IDEA 3

Modify Letter Cases

3. How can we modify the letter cases correctly?

Textbook problem!

(2024/25 C++ Foundation Course slides Chapter 10 Page 25-28)

For a character `S[i]`...

First, we need to identify its current case, we can use the condition

`S[i] >= 'A' && S[i] <= 'Z'` to identify if it is in **uppercase**

`S[i] >= 'a' && S[i] <= 'z'` to identify if it is in **lowercase**

KEY IDEA 3

Modify Letter Cases

3. How can we modify the letter cases correctly?

For a character `S[i]`...

First, we need to identify its current case, we can use the condition

`S[i] >= 'A' && S[i] <= 'Z'` to identify if it is in **uppercase**

`S[i] >= 'a' && S[i] <= 'z'` to identify if it is in **lowercase**

Then, to modify it

from lowercase to uppercase `c = c-'a'+'A'`

from uppercase to lowercase `c = c-'A'+'a'`

Subtasks

By having these three key ideas, we can solve the whole problem easily!

Here's an overview of the ideas involved in the subtasks:

Subtask	*Score*	S consists of...	1 Input with Spaces	2 Identify First Letter	3 Modify Letter Cases
1	*5*	`A` and `a`		☆	★
2	*10*	Lowercase letters		☆	☆
3	*15*	Letters		☆	★
4	*7*	`A`, `a` and spaces	★	★	☆
5	*5*	Lowercase letters and spaces	★	★	☆
6	*3*	Letters and spaces	★	★	★

Reflections

This problem is very simple.

The problem statement is simple.

The solution idea is simple.

The code is simple.

ALL knowledge required are in 2024/25 C++ Foundation Course slides Chapter 10, and they are very straightforward to identify and implement.

As long as you have done some revisions on it or print it on your 3-page reference paper, you will easily get **Accepted** in this problem!

5 | **15**	10 | **14**	15 | **14**	7 | **13**	5 | **13**	3 | **13**

Still somehow only 15 out 33 participants get points in this problem :(

Python Solution

```
print(input().title())
```

Result

Subtask	Verdict	Score
1	Accepted	5 /5
2	Accepted	10 /10
3	Accepted	15 /15
4	Accepted	7 /7
5	Accepted	5 /5
6	Accepted	3 /3

Tutor Assignment

TS254

Problem Statement

Given N participants M tutors

For the i^{th} participant,

it has name S_i

and is assigned to tutor $\#T_i$

For each tutor,

if no participant is assigned to him/her, output `UNEMPLOYED`

otherwise output a list of participants assigned to him/her

Subtask 1

$M = 1$

There is only one tutor.

→ Every participants will be assigned to tutor #1.

As $N \geq 1$, tutor #1 is never `UNEMPLOYED`.

We can just output all S_i on the same line, each separated by a space.

Expected Score	5	Cumulative	5

Subtask 2

$T_1 = T_2 = \cdots = T_N$

All participants are assigned to the same tutor.

When inputting each participant,

we can use `cin >> S[i] >> t`, where `t` is an `int`.

For outputting each tutor,

if the tutor # is equals to `t`, output all S_i,

otherwise output `UNEMPLOYED`.

Expected Score	**13**	Cumulative	13

Subtask 3

$N = M$
$T_1 \neq T_2 \neq \cdots \neq T_N$

Each tutor is assigned one participant.

The problem is technically the same as B012 Rank List.

We can have `string ans[]`.

For each participant,

`cin >> s >> t`, where `s` is a `string`, `t` is an `int`,

then we just save `ans[t] = s`.

Expected Score	10	Cumulative	23

Subtask 4

$1 \leq N, M \leq 5000$

For the `i`th participant, `cin >> S[i] >> T[i]`.

For the `i`th tutor,

loop `j` from `0` to `N-1`, output `S[j]` if `T[j]` equals to `i`,
if there's no matches, output `UNEMPLOYED`.

O(NM)	Expected Score	13	Cumulative	36

Full Solution

The Subtask 4 solution
loops through all N participants for each of the M tutors.

However, N and M can be as large as 10^5,
the Subtask 4 solution will take up to $10^5 \times 10^5 = 10^{10}$ operations.

Computer can only process around 10^9 operations within the 1s time limit, so the Subtask 4 solution will grant you Time Limit Exceeded for the whole problem.

Therefore, we need to have a faster solution.

Full Solution

We can build our full solution upon the Subtask 3 solution.

In Subtask 3, a tutor has only one participant assigned.

Thus we can create an array `string ans[]`.

For each participant,

`cin >> s >> t`, where `s` is a `string`, `t` is an `int`,

then we just save `ans[t] = s`.

But now a tutor can have multiple participants assigned,
how can we deal with this?

Full Solution

Instead of storing only one string for each element in the array, we can store multiple strings in each element in the array!

We can create an array of vector: `vector<string> ans[]`.

For each participant,

> `cin >> s >> t`, where `s` is a `string`, `t` is an `int`,
>
> then we push the participant's name into the correct place: `ans[t].push_back(s)`.

Full Solution

For each participant,

`cin >> s >> t`, where `s` is a `string`, `t` is an `int`,

then we push the participant's name into the correct place: `ans[t].push_back(s)`.

Finally for the i^{th} tutor,

if the size of `ans[i]` is 0, output `UNEMPLOYED`,

otherwise loop `j` from `0` from `ans[i].size()-1`,

output `ans[i][j]` separated by spaces.

O($N+M$)	Expected Score	**50**	Cumulative	50

Reflections

Subtasks are important.

Even if you don't know the full solutions, there are always (at least in our problems) some easy subtasks to do!

Look at the statement and sample tests to understand the problem.

Then view the subtasks to see which of them are possible.

In this problem, don't tell me you cannot solve Subtask 1!

5 | **11**	8 | **9**	10 | **9**	13 | **10**	14 | **9**

Sadly only 11 out 33 participants solved Subtask 1 :(

Reflections

This problem requires *some* thinking.

This kind of using C++ built-in data structures to solve problems is critical in beginners' level,

you will face many more problems similar to this in the near future,

and you will face much more difficult problems in your future OI journey.

In fact, you are recommended to try to do more problems (start from the easy ones to not-that-easy ones).

That's the most effective way (maybe the only way) for you to improve in coding.

Typical PTCG

TS255

Problem Statement

There are $2N$ cards, each card has a codeforce F_i.

In each round in the game, Alice picks one card and Bob picks one card after it.

The score of each player is the sum of codeforces of the cards he/she picks.

They will perform the best strategy in order to get the highest mark they can within the rule.

Calculate the final score difference.

Subtask 1

$N = 1$

There are 2 cards.

As Alice picks first, she will pick the card with the higher codeforce and Bob will pick another one.

The score difference can be calculated by $\max(F_1, F_2) - \min(F_1, F_2)$.

O(	1	)	Expected Score	6	Cumulative	6

Subtask 2

$1 \leq N \leq 4$

There are at most 8 cards.

Therefore, we can try all permutations to find out the answer.

We can use the function `next_permutation()` to simulate which card Alice will choose first, and simulate what will they choose next, finally calculate the final score difference at that permutation.

O(	$N!$	)	Expected Score	**14**	Cumulative	14

Observation 1

How can Alice perform the best strategy?

Choose the maximum codeforce in the cards.

How can Bob perform the best strategy?

Choose the second maximum codeforce in the cards.

Subtask 3

$F_1 \leq F_2 \leq F_3 \leq \cdots \leq F_{2N}$

Using Observation 1,
we can simply calculate the answer by adding
$F_{2N} - F_{2N-1}, F_{2N-2} - F_{2N-3}, \cdots, F_2 - F_1$.

O(N)	Expected Score	**12**	Cumulative	26

Subtask 4

$1 \leq N, F_i \leq 2000$

Now the codeforces haven't been sorted! How can we do?

Actually, we can simply run through all the cards to find the maximum and update the codeforce of the card become 0 to avoid re-adding.

Each time in the N times, finding the maximum, add it, set it to 0, then find the maximum again, subtract it, set it to 0.

O(N^2)	Expected Score	**13**	Cumulative	39

Subtask 5

As $1 \leq N \leq 10^5$, O(N^2) doesn't work anymore.

So we need to find the maximum each time by a more optimised method.

That is using sorting at the first!

After sorting, the codeforces of the cards will be sorted from the smallest to the largest.

Therefore, we can just apply the method we use in Subtask 3 to calculate the final score difference!

O(	$N \lg N$	)	Expected Score	**50**	Cumulative	50

Tick-tock Light

TS256

Problem Statement

There is a light switching on and off with the i^{th} interval between each switch is A_i.

Moreover, the switching pattern forms a loop where the $(N+j)^{\text{th}}$ interval between each switch is A_j.

As $256 = 2^8$, TS256 is a Maths problem.

Subtask 1

$N = 1$

The light switch for every A_1 seconds.

It is obvious that the light is on for A_1 seconds every $2A_1$ seconds.

Let $2A_1$ seconds be a period.

The answer is # of period $\times A_1 +$ answer in the incomplete period.

If the length of the incomplete period is larger than A_i seconds, the answer in the incomplete period is just A_i.

Otherwise, the whole period is light on.

Expected Score	**12**	Cumulative	12

Subtask 2

$T_1 + T_2 + \cdots + T_N \geq K$

Notice that the loop doesn't begin in the first K seconds.

However, simply looping K times doesn't work as K can be as large as 10^9.

We can observe that the light is on in the first T_1 seconds.

Instead of looping T_1 times, we can directly adding the answer by T_1.

Similarly, T_2 is omitted and T_3 is added to the answer.

Carefully handle the case where you have less than T_i seconds left.

Expected Score	**13**	Cumulative	25

Subtask 3

$1 \leq K \leq 10^5$

As K is small, we can loop for each seconds, checking whether it is on or off.

For the i^{th} interval, we can get the length by $A[(i-1)\%N+1]$.

Expected Score	14	Cumulative	39

Subtask 4

N is even

Let combine the idea of Subtask 1 and 2.

Let a period be $T_1 + T_2 + \cdots + T_N$.
As N is even, for each period the light will on for $t = T_1 + T_3 + \cdots + T_{N-1}$.
The answer is # of period $\times\ t +$ answer in the incomplete period.

How can we calculate the answer in the complete period?
Subtask 2!

Expected Score	**15**	Cumulative	54

Full Solution

For the full problem, N can be odd.

Similar to Subtask 1, we can double the size of the period.

When N is odd, let a period be $2 \times (T_1 + T_2 + \cdots + T_N)$ seconds, the light will on for $t = T_1 + T_2 + \cdots + T_N$ seconds.

Subtask4!

Expected Score	65	Cumulative	65

Trivial Colour

TS257

Problem Statement

Each colour can be represented by a triple (R, G, B)
where $0 \leq R, G, B \leq 255$.

When two colours (R_1, G_1, B_1) and (R_2, G_2, B_2) are mixed together,
the result colour can be represented by $(\lfloor \frac{R_1+R_2}{2} \rfloor, \lfloor \frac{G_1+G_2}{2} \rfloor, \lfloor \frac{B_1+B_2}{2} \rfloor)$.

There are Q queries where
each query ask you the mixed colour of two given colours.

Subtask 1

$N = 3$
$R_i = G_i = B_i$

Notice that there is only 3 colours and each colour can be represented by an integer.

To calculate the mixed colour, we need to first convert the colour from its name to its corresponding integer.

How?

Use a series of ifs.

Subtask 1

Let the integer representations of the two colours be X and Y.

The result colour can be represented by $S = \lfloor \frac{X+Y}{2} \rfloor$.

How can we convert the integer representation to its name?

Use a series of ifs.

Reminded to output `TRIVIAL` if it isn't one of the three colours.

O(Q)	Expected Score	**18**	Cumulative	18

Subtask 2

$R_i = G_i = B_i$

Hidden constraint: $1 \leq N \leq 256$ as colours are unique

N is much larger now.

Can we use a series of ifs again?

Yes!!!

However, there is a better way to do so.

Declaring 512 variables and writing 512 ifs are too risky.

Any mistake cause a zero mark.

Subtask 2

Instead, we can use an array to save all elements.

To convert a colour from its name to its integer representation, use for loop:

If name is equal to C_i

`tmp` ← R_i

break

`tmp` will be the integer representation of the colour.

Similarly, you can convert the integer representation to its name.

O(NQ)	Expected Score	**38**	Cumulative	38

Subtask 3

$1 \leq N, Q \leq 1000$
$0 \leq R_i, G_i, B_i \leq 63$

R_i, G_i, B_i are no longer equal, how can we store the name of the colours?

Use a 3D array instead of a simple array!

As NQ is small, we can still use a for loop to convert.

O(NQ)	Expected Score	17	Cumulative	55

Subtask 4

$0 \leq R_i, G_i, B_i \leq 63$

NQ can be as large as 10^{10} now, you will receive TLE using simple loop.

To convert an integer representation to its name, we can just put the integers to the index of the 3D array.

```
name = arr[R_i][G_i][B_i]
```

Remember to initialise the array to check if the colour exists.

Subtask 4

To convert a name to its integer representation,
We can use a data structure in C++ standard library.
Your best friend – `map`.
Using `map`, we can put string as key with a cost of $O(\lg N)$.

Refer to 2024/25 C++ Foundation Course slides Chapter 9 for details.

$O(\ Q \lg N\ )$	Expected Score	**32**	Cumulative	70

Full Solution

Runtime error!!!

Why?

Notice that you have only 256MB.

Declaring a 3D array of string with size $256 \times 256 \times 256$ will use more than 256MB.

Instead we can make a structure of integer representation,

using map with the key as the integer representation can solve the problem.

Memory complexity: O($N \lg N$)

O($Q \lg N$)	Expected Score	**80**	Cumulative	80

Travelling Wonder

TS258

Problem Statement

There is a graph with N nodes and M edges.

Each edge connecting a pair of nodes with a weight of W.

There are Q queries.

For each query, you have to determine whether it is possible to travel from node u to node v only using edges with weight $\leq w$.

Subtask 1

$N = 2$

Hidden constraint: $M = 1$

When can you travel from node 1 to node 2?
Only if you can use the only edge!

Just use an if statement + a for loop will give you this easy 10 marks!

O(Q)	Expected Score	**10**	Cumulative	10

Subtask 2

$Q = 1$

There is only one scenario and you can only use edges with weight smaller than w.

Just build a graph with those edges and run BFS / DFS from node u.

O($N + M$)	Expected Score	**21**	Cumulative	31

Subtask 3

$1 \leq N, M, Q \leq 100$

Both N, M, Q are small, can we brute force?

Yes, build a new graph for every Q.

Run BFS / DFS again.

Writing some functions can definitely help you.

O($\boldsymbol{NMQ}$ / $\boldsymbol{(N+M)Q}$)	Expected Score	**24**	Cumulative	55

Subtask 4

$M = N - 1$
$U_i = i$
$V_i = i + 1$
$u_i = 1$

The graph is a simple chain and you always start at the one end.

Notice that there is only one way to travel from node 1 to node v.
We must travel through the first $v - 1$ edges.
If $w \geq \max(W_i)$ where $1 \leq i < v$, it is possible.
Otherwise, it is always impossible.

Subtask 4

However, Q is as large as 10^5, how can we query $\max(W_i)$ faster?

Precompute the prefix maximum.
Let $pm[v] = \max(W_i)$ where $1 \leq i < v$.
As $\max(W_i) = \max(\max(W_j), W_{v-1})$ where $1 \leq i < v$ and $1 \leq j < v - 1$,
we can observe that $pm[v] = \max(pm[v-1], W_{v-1})$.
For simple calculation, we can define $pm[1] = 0$.

Precompute the prefix maximum and we can use the result in O(1).

O($N + Q$)	Expected Score	**19**	Cumulative	74

Subtask 5

$M = N - 1$
$U_i = 1$
$V_i = i + 1$

Another special graph, it is a star with centre as node 1.

By comparing the scores of Subtask 4 and Subtask 5, star graph is actually easier to solve.

As the diameter of a star graph is 2, we can always travel from any node to any node using less than 3 edges.

Subtask 5

We can branch the case and solve them one by one.

Case 1: either $u = 1$ or $v = 1$

We can travel using only one edge.

Simple if statement.

Case 2: $u \neq v \neq 1$

We can travel using only two edges.

Simple if statement with help of `max()`.

O(Q)	Expected Score	**16**	Cumulative	90

Subtask 6

$1 \leq W_i \leq 2$

Now, both N, M, Q are large. We cannot run Q BFS / DFS.

Instead of thinking traveling from u to v, we can think whether u, v are connected.

How to check if two nodes are connected?

DSU!

Using DSU, we can query connectivity of two nodes in about $O(1)$.

Subtask 6

There are only two scenarios, $w = 1$ or $w > 1$.

If $w = 1$, we can only use edges with weight 1.
Check connectivity using a DSU.

If $w > 1$, we can use all edges.
Check connectivity using another DSU.

You are recommended to write a structure of DSU.

O($N + M + Q$)	Expected Score	**22**	Cumulative	112

Full Solution

There is a total of $M + 1$ ranges of w make a difference.

However, it is impossible to maintain $M + 1$ DSU.

We can observe that if we can edges with weight $\leq w$.

These edges must be able to use when we can edges with weight $\leq w + 1$.

We can get the second DSU base on the first DSU, again and again.

Full Solution

However, the order of the DSU we are going to use is not sorted.

As the query isn't online, we can first sort all queries by w.
We will never use less edges in this order.
Hence, we can maintain a DSU with all edges below w.
Using two pointers to mark which edges are going to be inserted.

Remember to sort the queries to the original order before outputting.

$O(N + M \lg M + Q \lg Q)$	Expected Score	**135**	Cumulative	135