香港電腦奧林匹克競賽
Hong Kong Olympiad in Informatics

G261 - Sushi Go

Isaac Wong {WongChun1234}
2026-02-14

Table of Contents

Background

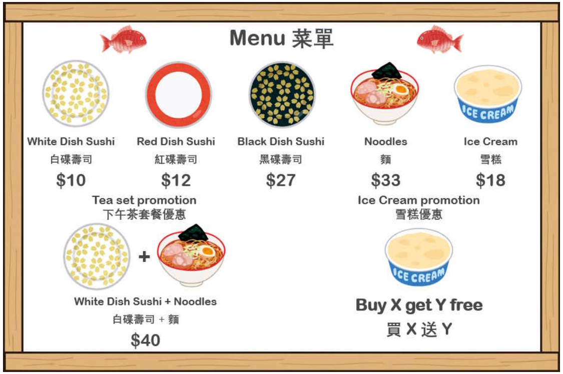
Problem Idea by `WongChun1234`
Preparation by `WongChun1234`

This problem asks you to compute the minimum cost of purchasing items in a sushi shop, given some discounts.

Rant from author: A sushi shop brand opened a new sushi shop near my home, but the queue is still very long :(

Abridged Statement

Find the minimum cost of purchasing items in a sushi shop, according to the following menu:



Abridged Statement

Special promotions provided:

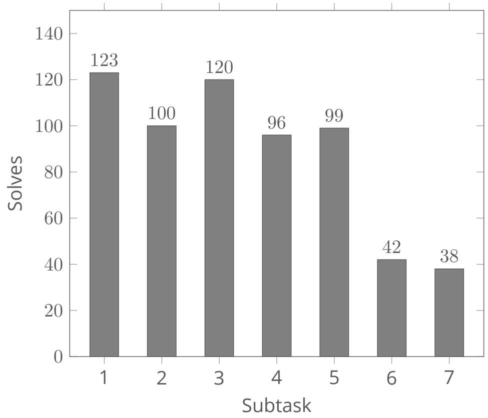
1. Tea Set Promotion: A tea set sold at a price $\$40$, which includes one White Dish Sushi and one bowl of Noodles.
   Condition: Only available from `14:00` to `16:59`
2. Ice Cream Promotion: There is a buy X get Y free promotion for Ice Cream.
   Condition: Only available when $Y > 0$

Subtasks

Subtask	Points	Promotion		A
		Tea Set	Ice Cream	
1	9	✗	✗	$= \{0, 2, 0, 1, 1\}$
2	9	✓	✓✗	$= \{2, 1, 1, 1, 3\}$
3	15	✗	✗	?
4	16	?	✗	?
5	18	✗	✓✗	?
6	25	✗	?	
7	8	?		

✓✗: Guaranteed to be buy 1 get 1 free.

Statistics



First solved by **as20191072** (Lei Yirui) at **13m 17s**.

Table of Contents

Subtask 1-2: The ordered items are given

In these subtasks, you are given the list of items Alice and Bob want and also the promotions available. You can calculate the minimum cost externally using pen and paper, then directly output the answer in your code.

In the first subtask, the answer is 57.
In the second subtask, the answer is 125.

Hello, World! Program Structure (C++)

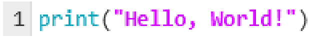
Let’s see how we can write a simple program to print the string “`Hello, World!`”.

If you selected your language as C++20 and opened the “Code” function, you should see the following template popping up:

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    cout << "Hello, World!" << endl;
    return 0;
}
```

Hello, World! Program Structure (Python)

Let’s see how we can write a simple program to print the string “`Hello, World!`”.
If you selected your language as Python 3 and opened the “Code” function, you should see the following template popping up:



```python
print("Hello, World!")
```

Subtask 1-2: The ordered items are given

Replace "`Hello, World!`" with whatever item we like to print. If we would like to print an integer, the quotation marks (" and ") are optional.

```cpp
#include <bits/stdc++.h>
using namespace std;
int main() {
    cout << 57 << endl;
    return 0;
}
```

C++ implementation (Subtask 1)

```python
print(125)
```

Python implementation (Subtask 2)

Expected score: 9 + 9 = 18

Note on cumulative scoring

In most OI contests, you can submit to the same problem multiple times. If cumulative scoring per subtask is enabled, your highest-scoring submission will be counted for each subtask individually. For example, if you pass subtask 1 in the first submission, and pass subtask 2 in the second submission, you will get the score of both subtasks.

Subtask	Submission 1	Submission 2	Total Score
1	9	0	9
2	0	9	9
3	0	0	0

Basic Contest Strategy

Since cumulative scoring per subtask is enabled in HKOI and HKGOI, you are encouraged to submit your code for easier subtasks before further complicating your code to try and solve the harder subtasks.

In this way, you can guarantee to have some marks from the easier subtasks, even if you cannot solve any more subtasks in the future or if you submit a completely wrong solution afterwards.

You can also spot any problems in your code earlier, such as misunderstanding of the problem, if you see that your code is not even passing the first subtask.

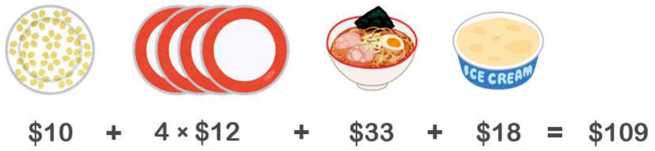
Table of Contents

Subtask 3: Neither special promotions are active

Since $H : M = 12{:}00$, and $Y = 0$, both special promotions are not active.

To calculate the minimum cost, we sum up the product of item price and number of items needed for each item. That is, the minimum cost is $A_1 \times 10 + A_2 \times 12 + A_3 \times 27 + A_4 \times 33 + A_5 \times 18$.

Example (Sample 1):



Inputting an Integer (C++)

We first declare an integer variable, then we input it using `cin >>` (opposite of `cout <<` for output).

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    int a1;
    cin >> a1;
    return 0;
}
```

Note: There are restrictions for variable naming, for example, a variable’s name cannot start with a digit, it cannot be equal to a keyword in C++, and more. Generally, it is safe to use single-letter variable names.

Inputting an Integer (Python)

Here is how you can read an integer on a separate line in Python. (The case where there are multiple integers on the same line is not as simple.)

```python
a1 = int(input())
```

The outer `int()` converts the input from a string into an integer – it is necessary, otherwise the input will be treated as a string!

Inputting Multiple Integers

You can simply copy and paste the same code 5 times to input all 5 integers. However, the code is quite long with low expandability. (Imagine if we needed to input 500 integers!) We could use an array and for-loops instead.

```cpp
vector<int> a(5);
for (int i = 0; i < 5; i++) {
    cin >> a[i];
}
```

C++ implementation

```python
a = [int(input()) for _ in range(5)]
```

Python implementation

Don't worry if they look too complicated. Using them is optional, and you will learn them later.

Subtask 3: Neither special promotions are active

We can also use the for-loop and constant array to calculate the total cost.
Example implementations:

```cpp
#include <bits/stdc++.h>
using namespace std;
int main() {
    vector<int> a(5);
    for (int i = 0; i < 5; i++) {
        cin >> a[i];
    }
    vector<int> cost = {10, 12, 27, 33, 18};
    int ans = 0;
    for (int i = 0; i < 5; i++) {
        ans += a[i] * cost[i];
    }
    cout << ans << endl;
    return 0;
}
```

C++ implementation

```python
a = [int(input()) for _ in range(5)]
cost = [10, 12, 27, 33, 18]
ans = 0
for i in range(5):
    ans += a[i] * cost[i]
print(ans)
```

Python implementation

```python
a = [int(input()) for _ in range(5)]
cost = [10, 12, 27, 33, 18]
ans = sum(x * y for x, y in zip(a, cost))
print(ans)
```

Python implementation (advanced)

Expected score: 24 (Cumulative: 33)

Table of Contents

Subtask 4: Ice cream promotion is not active

In this subtask, we need to check two additional things.

- Check whether tea set promotion is active or not
- If tea set promotion is active, calculate the cost saved using the tea set

Subtask 4: Ice cream promotion is not active

1. Check whether tea set promotion is active or not

This part is the easier part. As the tea set promotion is active from `14:00` to `16:59`, we only need to check if H is between 14 and 16 (inclusive) or not. In fact, we do not need to use the value of M at all in this task.

To do this we need to be familiar with the syntax of **if-statements**. Essentially, if the `condition` (in the code shown below) holds (is **true**), then run the code in the "execute code here" section.

```cpp
if (condition) {
    //execute code here
}
```

C++ if-statement

```python
if condition:
    #execute code here
```

Python if-statement

Subtask 4: Ice cream promotion is not active

1. Check whether tea set promotion is active or not

However, there are slight problems for both C++ and Python. H and W are on the same line in the input. In C++, we can input them normally using two `cin`. However, `input` in python reads the whole line. We need to `split` it into multiple strings, then convert them into integers individually.

```cpp
cin >> h >> w;
```

C++

```python
h, w = input().split()
h = int(h)
w = int(w)
```

Python

```python
tmp = list(map(int, input().split()))
h = tmp[0]
w = tmp[1]
```

Python (array implementation)

In Python, we can check the condition directly using `14 <= H <= 16`. However this is invalid syntax in C++. We need to separate it into `H >= 14 && H <= 16` (`&&` is the boolean **AND** operator).

Subtask 4: Ice cream promotion is not active

2. If tea set promotion is active, calculate the cost saved using the tea set

The cost of buying one white dish sushi and one bowl of noodles individually is $\$10 + \$33 = \$43$, which is higher than the cost of the tea set $\$40$. By ordering the tea set we can save $\$43 - \$40 = \$3$.
However, it is not worth ordering the tea set if we only want one of the items, as $\$40 > \$10$ and $\$40 > \$33$.

Subtask 4: Ice cream promotion is not active

2. If tea set promotion is active, calculate the cost saved using the tea set

Therefore, the number of tea sets ordered should be the **minimum** of the number of white dish sushi ordered and the bowls of noodles ordered.
Example (Sample 2):

$40 + $40 + $10 + $10 = $100

As they want 4 white dish sushi and 2 bowls of noodles, they should order $\min(4, 2) = 2$ tea sets.

Subtask 4: Ice cream promotion is not active

2. If tea set promotion is active, calculate the cost saved using the tea set

We can save $\$3 \times \min(A_1, A_4)$ from the tea set promotion if it is active. We can use the `min` function directly in both C++ and Python, as it is included by `#include <bits/stdc++.h>` in C++ and built-in in python.

IMPORTANT: Even though in the task the items are numbered from 1 to 5 (1-based), in both C++ and Python, an array of size 5 is indexed from 0 to 4 (0-based). Therefore we should actually reduce the answer by `3 × min(A[0], A[3])` if we wish to use the array approach. In the following slides, array A with subscript (e.g. A_5) are 1-based (as in the problem statement), while array `A` with square brackets (e.g. `A[4]`) are 0-based (as in the C++ / Python code).

Subtask 4: Ice cream promotion is not active

```cpp
cin >> h >> w

//omitted

if (h >= 14 && h <= 16) {
	ans -= 3 * min(a[0], a[3]);
}
```

C++ implementation

```python
h, w = input().split()
h = int(h)

#omitted

if (14 <= h <= 16):
	ans -= 3 * min(a[0], a[3])
```

Python implementation

Expected score: 40 (Cumulative: 49)

Table of Contents

Subtask 5: Only buy 1 get 1 free ice cream promotion is active.

Now, we need to deal with the ice cream promotion. What does buy 1 get 1 free for ice cream actually mean?

ICE CREAM
paid
free
paid
free
paid
free
. . .

We can group the ice cream into pairs. In each pair, the first one is paid and the second one is free. Thus, the number of free cups of ice cream is $\lfloor A_5/2 \rfloor$.

- When $A_5 = 5$, we can get $\lfloor \frac{5}{2} \rfloor = \lfloor 2.5 \rfloor = 2$ cups of ice cream for free.
- When $A_5 = 6$, we can get $\lfloor \frac{6}{2} \rfloor = \lfloor 3 \rfloor = 3$ cups of ice cream for free.

Note: when A_5 is odd, we actually get an extra cup of ice cream. However, we still need to pay for $A_5 - \lfloor \frac{A_5}{2} \rfloor$ cups of ice cream to get $A_5 + 1$ cups of ice cream.

Subtask 5: Only buy 1 get 1 free ice cream promotion is active.

In C++, the division operator / is floored division. That is, `X / Y` in C++ results in $\lfloor \frac{X}{Y} \rfloor$. In Python, the division operator / results in a floating point number. That is, `X / Y` in Python might not result in an integer. To calculate floored division in Python, we should use `X // Y`.
Remember to use `A[4]` if you are using the array implementation, and calculate $\lfloor \frac{A_5}{2} \rfloor$ first before multiplying it by $\$18$.

```cpp
ans -= 18 * (a[4] / 2);
```

C++ implementation

```python
ans -= 18 * (a[4] // 2)
```

Python implementation

Expected score: 27 (Cumulative: 67)

Subtask 6: Tea set promotion is not active

Now, let us consider a "Subtask 5.5", where $Y = 1$. For simplicity, we will use P and F to represent paid and free ice cream respectively. We will also use C to represent the number of cups of ice cream ordered and underline them.

1. $X = 1$, $C = 3$: PF / PF / PF / ...
2. $X = 3$, $C = 8$: PPPF / PPPF / PPPF / ...
3. $X = 4$, $C = 7$: PPPPF / PPPPF / ...

Can you find the expression to calculate the number of cups of ice cream we can get for free?

Subtask 6: Tea set promotion is not active

When $Y = 1$, we get $\lfloor \frac{C}{X+1} \rfloor$ (or alternatively $\lfloor \frac{C}{X+Y} \rfloor$) cups of ice cream for free.

What about when $Y = 2$?

1. $X = 1, C = 6$: PFF / PFF / PFF / ...
2. $X = 3, C = 8$: PPPFF / PPPFF / PPPFF / ...
3. $X = 2, C = 7$: PPFF / PPFF / ...

Do we still get $2 \times \lfloor \frac{C}{X+Y} \rfloor$ cups of ice cream for free?

No! Consider the third case. $2 \times \lfloor \frac{7}{2+2} \rfloor = 2 \times \lfloor 1.75 \rfloor = 2 \times 1 = 2$, but we actually get 3 cups of ice cream for free. (In fact, we actually get 4 cups of ice cream for free for a total of 8 cups of ice cream, but let's assume that we discard that extra cup of ice cream.)

Subtask 6: Tea set promotion is not active

$X = 2, Y = 2, C = 7$: PPFF / PPFF / ...
For the first 4 cups of ice cream, we handled the buy 2 get 2 free promotion correctly. However, in the last 3 cups of ice cream, we are effectively using a "buy 2 get 1 free" promotion. This error occurs as $X = 2 < 3 < X + Y = 4$, where we did not fully use the promotion. We can look at another case.

$X = 2, Y = 3, C = 13$: PPFFF / PPFFF / PPFFF / ...
In this case, the last 3 cups of ice cream face the same error as the previous case, where $X = 2 < 3 < X + Y = 5$.

Subtask 6: Tea set promotion is not active

$X = 2, Y = 2, C = 7$: PPFF / PPFF / ...

To fix this, let R be the remainder of C divided by $X + Y$. We then compare R to X, as R is always smaller than $X + Y$. When $R > X$, we get $R - X$ extra cups of ice cream for free. In this case, $R = 3$, so we get $R - X = 3 - 2 = 1$ extra cup of ice cream for free.

In both C++ and Python, you can calculate the remainder R by using the modulo operator %. The remainder of X divided by Y is `X % Y`.

Subtask 6: Tea set promotion is not active

Implementation notes: We can use the expression max$(0, R - X)$ to determine the extra cups of ice cream we can get for free, as $R - X < 0$ implies $R < X$, which we get no more ice cream for free. Also, this expression can handle $Y = 0$ as well since $X \geq 1$ (calculating `C % 0` will result in a runtime error). Always remember to use `A[4]` instead of A_5!

```cpp
ans -= 18 * y * (a[4] / (x + y));
ans -= 18 * max(0, a[4] % (x + y) - x);
```

C++ implementation

```python
ans -= 18 * y * (a[4] // (x + y))
ans -= 18 * max(0, a[4] % (x + y) - x)
```

Python implementation

Expected score: 67 (Cumulative: 92) (Almost there!)

Table of Contents

Subtask 7: No additional constraints

It remains to combine the solutions to subtask 4 (tea set promotion) and subtask 6 (ice cream promotion).
Solution outline:

1. Input array A (subtask 3)
2. Input H, W, X, Y (subtask 4)
3. Handle base cost (subtask 3)
4. Handle tea set promotion (subtask 4)
5. Handle ice cream promotion (subtask 6)
6. Output the answer (subtask 1-2)

Expected score: 100 :)
You should be able to code it out yourself now. Try merging the subtasks yourself and refer to the previous slides when needed!

Aside: Other Implementations

In fact, there are many ways you can write a program to solve this task, some of which may be *better* than others... (by better, we mean cleaner/neater code, more convenient code, code that takes less time to write, etc). For example:

- Calculate the cost directly instead of reducing it for the promotions
- Alternative equations for calculating the promotions
- Directly simulate the promotions! This is possible as elements in A are small.

There isn't a single *correct* way to solve a problem. Learn from others' code and you may improve from it!

Table of Contents

What's next?

- We have almost reached the end of this presentation.
- We have gotten through bits and pieces of basic C++/Python syntax, and used them to gradually solve each subtask.
- Now, it is a good idea to familiarize yourself with the syntax. Then, try to come up with the solution to this task *without directly copying the previous slides* or *looking up any documentation*!
- Submit your program on HKOI Online Judge, and you will know whether your implementation is correct or not. If you get stuck, depending on what score you get, come back to the slides to check for potential bugs.

What's next?

- To some of you, this may be the first OI/Competitive Programming task you have encountered/solved. Congratulations!
- After solving this task, you are highly encouraged to **upsolve** other tasks as well *(to solve a problem after the end of a contest)* :)
- Also don't hesitate to solve other problems on HKOI Online Judge / other online judges to improve your coding and problem solving skills.
- The HKOI training team also holds weekly training on Saturdays, roughly from February to May. You can chat with trainers directly there and improve a lot from it. You may join if you are a HKGOI medalist / HKOI finalist.

Some Resources

- HKOI Online Judge **(recommended – probably your best friend for now!)**
  - DSE exercises to practice basic syntax
  - Some easy problems for competitive programming beginners
  - More resources to be released in the future… (stay tuned!)
- Codeforces
  - Regular contests (recommended after you are familiar with basic syntax)
- AtCoder
  - Regular contests (recommended after you are familiar with basic syntax)

The End

- Good luck in your OI journey!
- Start/Keep practicing, and never give up.