Dhaka International University

Submit By ,

Name : Md. Moinul Islam Sajid.

Batch : 74th.

Roll No. : 19.

HACAR RANK

Gmail: [moinulislamsajid107@gmail.com](mailto:moinulislamsajid107@gmail.com)

* Pass : Sajid10

Subject : CSE – 103 And CSE Lab.

Assignment 01

Submit to,

Name: Mahmudul Hasan.

Lecturer

Dept. Of CSE

Dhaka International University

Problem 1 :

1. "Hello World!" in C

**Sample Input 0**

Welcome to C programming.

**Sample Output 0**

Hello, World!

Welcome to C programming.

Problem 2 :

The fundamental data types in c are int, float and char. Today, we're discussing int and float data types.

The printf() function prints the given statement to the console. The syntax is printf("format string",argument\_list);. In the function, if we are using an integer, character, string or float as argument, then in the format string we have to write %d (integer), %c (character), %s (string), %f (float) respectively.

The scanf() function reads the input data from the console. The syntax is scanf("format string",argument\_list);. For ex: The scanf("%d",&number) statement reads integer number from the console and stores the given value in variable .

To input two integers separated by a space on a single line, the command is scanf("%d %d", &n, &m), where  and  are the two integers.

**Task**

Your task is to take two numbers of int data type, two numbers of float data type as input and output their sum:

1. Declare  variables: two of type int and two of type float.
2. Read  lines of input from stdin (according to the sequence given in the 'Input Format' section below) and initialize your  variables.
3. Use the  and  operator to perform the following operations:
   * Print the sum and difference of two int variable on a new line.
   * Print the sum and difference of two float variable rounded to one decimal place on a new line.

**Input Format**

The first line contains two integers.  
The second line contains two floating point numbers.

**Constraints**

* integer variables
* float variables

**Output Format**

Print the sum and difference of both integers separated by a space on the first line, and the sum and difference of both float (scaled to  decimal place) separated by a space on the second line.

**Sample Input**

10 4

4.0 2.0

**Sample Output**

14 6

6.0 2.0

Problem 3 : **Objective**

In this challenge, you will learn simple usage of functions in C. Functions are a bunch of statements grouped together. A function is provided with zero or more arguments, and it executes the statements on it. Based on the return type, it either returns nothing (void) or something.  
  
A sample syntax for a function is

return\_type function\_name(arg\_type\_1 arg\_1, arg\_type\_2 arg\_2, ...) {

...

...

...

[if return\_type is non void]

return something of type `return\_type`;

}

For example, a function to read four variables and return the sum of them can be written as

int sum\_of\_four(int a, int b, int c, int d) {

int sum = 0;

sum += a;

sum += b;

sum += c;

sum += d;

return sum;

}

+= : Add and assignment operator. It adds the right operand to the left operand and assigns the result to the left operand.

a += b is equivalent to a = a + b;

**Task**

Write a function int max\_of\_four(int a, int b, int c, int d) which reads four arguments and returns the greatest of them.

**Note**

There is not built in max function in C. Code that will be reused is often put in a separate function, e.g. int max(x, y) that returns the greater of the two values.

**Input Format**

Input will contain four integers -  , one on each line.

**Output Format**

Print the greatest of the four integers.  
Note: I/O will be automatically handled.

**Sample Input**

3

4

6

5

**Sample Output**

6

Problem 4: **Objective**

In this challenge, you will learn to implement the basic functionalities of pointers in C. A [pointer](http://en.wikipedia.org/wiki/Pointer_%28computer_programming%29) in C is a way to share a memory address among different contexts (primarily functions). They are primarily used whenever a function needs to modify the content of a variable that it does not own.  
  
In order to access the memory address of a variable, , prepend it with  sign. For example, &val returns the memory address of .  
  
This memory address is assigned to a pointer and can be shared among various functions. For example,  will assign the memory address of  to pointer . To access the content of the memory to which the pointer points, prepend it with a \*. For example, \*p will return the value reflected by  and any modification to it will be reflected at the source ().

void increment(int \*v) {

(\*v)++;

}

int main() {

int a;

scanf("%d", &a);

increment(&a);

printf("%d", a);

return 0;

}

**Task**

Complete the function void update(int \*a,int \*b). It receives two integer pointers, int\* a and int\* b. Set the value of  to their sum, and  to their absolute difference. There is no return value, and no return statement is needed.

**Input Format**

The input will contain two integers,  and , separated by a newline.

**Output Format**

Modify the two values in place and the code stub main() will print their values.

Note: Input/ouput will be automatically handled. You only have to complete the function described in the 'task' section.

**Sample Input**

4

5

**Sample Output**

9

1

**Explanation**

Problem 5: **Objective**

*if* and *else* are two of the most frequently used conditionals in C/C++, and they enable you to execute zero or one conditional statement among many such dependent conditional statements. We use them in the following ways:

1. if: This executes the body of bracketed code starting with  if  evaluates to *true*.
2. if (condition) {
3. statement1;
4. ...
5. }
6. if - else: This executes the body of bracketed code starting with  if  evaluates to *true*, or it executes the body of code starting with  if  evaluates to *false*. Note that only *one* of the bracketed code sections will ever be executed.
7. if (condition) {
8. statement1;
9. ...
10. }
11. else {
12. statement2;
13. ...
14. }
15. if - else if - else: In this structure, dependent statements are chained together and the  for each statement is only checked if all prior conditions in the chain are evaluated to *false*. Once a  evaluates to *true*, the bracketed code associated with that statement is executed and the program then skips to the end of the chain of statements and continues executing. If each  in the chain evaluates to false, then the body of bracketed code in the *else* block at the end is executed.
16. if(first condition) {
17. ...
18. }
19. else if(second condition) {
20. ...
21. }
22. .
23. .
24. .
25. else if((n-1)'th condition) {
26. ....
27. }
28. else {
29. ...
30. }

**Task**

Given a positive integer denoting , do the following:

* If , print the lowercase English word corresponding to the number (e.g., one for , two for , etc.).
* If , print Greater than 9.

**Input Format**

The first line contains a single integer, .

**Constraints**

**Output Format**

If , then print the lowercase English word corresponding to the number (e.g., one for , two for , etc.); otherwise, print Greater than 9 instead.

**Sample Input**

5

**Sample Output**

five

**Sample Input #01**

8

**Sample Output #01**

eight

**Sample Input #02**

44

**Sample Output #02**

Greater than 9

Problem 6: **Objective**

In this challenge, you will learn the usage of the *for* loop, which is a programming language statement which allows code to be executed until a terminal condition is met. They can even repeat forever if the terminal condition is never met.

The syntax for the for loop is:

for ( <expression\_1> ; <expression\_2> ; <expression\_3> )

<statement>

* *expression\_1* is used for intializing variables which are generally used for controlling the terminating flag for the loop.
* *expression\_2* is used to check for the terminating condition. If this evaluates to false, then the loop is terminated.
* *expression\_3* is generally used to update the flags/variables.

The following loop initializes  to 0, tests that  is less than 10, and increments  at every iteration. It will execute 10 times.

for(int i = 0; i < 10; i++) {

...

}

**Task**

For each integer  in the interval  (given as input) :

* If , then print the English representation of it in lowercase. That is "one" for , "two" for , and so on.
* Else if  and it is an even number, then print "even".
* Else if  and it is an odd number, then print "odd".

**Input Format**

The first line contains an integer, .  
The seond line contains an integer, .

**Constraints**

**Output Format**

Print the appropriate English representation,even, or odd, based on the conditions described in the 'task' section.

**Note:**

**Sample Input**

8

11

Problem 7: **Objective**

The modulo operator, %, returns the remainder of a division. For example, 4 % 3 = 1 and 12 % 10 = 2. The ordinary division operator, /, returns a truncated integer value when performed on integers. For example, 5 / 3 = 1. To get the last digit of a number in base 10, use  as the modulo divisor.

**Task**

Given a five digit integer, print the sum of its digits.

**Input Format**

The input contains a single five digit number, .

**Constraints**

**Output Format**

Print the sum of the digits of the five digit number.

**Sample Input 0**

10564

**Sample Output 0**

16

Problem 8: In this challenge, you will use logical bitwise operators. All data is stored in its binary representation. The logical operators, and C language, use  to represent true and  to represent false. The logical operators compare bits in two numbers and return true or false,  or , for each bit compared.

* Bitwise AND operator & The output of bitwise AND is *1* if the corresponding bits of two operands is *1*. If either bit of an operand is *0*, the result of corresponding bit is evaluated to *0*. It is denoted by &.
* Bitwise OR operator | The output of bitwise OR is *1* if at least one corresponding bit of two operands is *1*. It is denoted by |.
* Bitwise XOR (exclusive OR) operator ^ The result of bitwise XOR operator is *1* if the corresponding bits of two operands are opposite. It is denoted by .

For example, for integers 3 and 5,

3 = 00000011 (In Binary)

5 = 00000101 (In Binary)

AND operation OR operation XOR operation

00000011 00000011 00000011

& 00000101 | 00000101 ^ 00000101

\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_

00000001 = 1 00000111 = 7 00000110 = 6

You will be given an integer , and a threshold, i1nnik$. Print the results of the and, or and exclusive or comparisons on separate lines, in that order.

**Example**

The results of the comparisons are below:

a b and or xor

1 2 0 3 3

1 3 1 3 2

2 3 2 3 1

For the and comparison, the maximum is . For the or comparison, none of the values is less than , so the maximum is . For the xor comparison, the maximum value less than  is . The function should print:

2

0

2

**Function Description**

Complete the *calculate\_the\_maximum* function in the editor below.

*calculate\_the\_maximum* has the following parameters:

* *int n:* the highest number to consider
* *int k:* the result of a comparison must be lower than this number to be considered

**Prints**

Print the maximum values for the and, or and xor comparisons, each on a separate line.

**Input Format**

The only line contains  space-separated integers,  and .

**Constraints**

**Sample Input 0**

5 4

**Sample Output 0**

2

3

3

Problem 9: Print a pattern of numbers from  to  as shown below. Each of the numbers is separated by a single space.

4 4 4 4 4 4 4

4 3 3 3 3 3 4

4 3 2 2 2 3 4

4 3 2 1 2 3 4

4 3 2 2 2 3 4

4 3 3 3 3 3 4

4 4 4 4 4 4 4

**Input Format**

The input will contain a single integer .

**Constraints**

**Sample Input 0**

2

**Sample Output 0**

2 2 2

2 1 2

2 2 2

**Sample Input 1**

5

**Sample Output 1**

5 5 5 5 5 5 5 5 5

5 4 4 4 4 4 4 4 5

5 4 3 3 3 3 3 4 5

5 4 3 2 2 2 3 4 5

5 4 3 2 1 2 3 4 5

5 4 3 2 2 2 3 4 5

5 4 3 3 3 3 3 4 5

5 4 4 4 4 4 4 4 5

5 5 5 5 5 5 5 5 5

**Sample Input 2**

7

**Sample Output 2**

7 7 7 7 7 7 7 7 7 7 7 7 7

7 6 6 6 6 6 6 6 6 6 6 6 7

7 6 5 5 5 5 5 5 5 5 5 6 7

7 6 5 4 4 4 4 4 4 4 5 6 7

7 6 5 4 3 3 3 3 3 4 5 6 7

7 6 5 4 3 2 2 2 3 4 5 6 7

7 6 5 4 3 2 1 2 3 4 5 6 7

7 6 5 4 3 2 2 2 3 4 5 6 7

7 6 5 4 3 3 3 3 3 4 5 6 7

7 6 5 4 4 4 4 4 4 4 5 6 7

7 6 5 5 5 5 5 5 5 5 5 6 7

7 6 6 6 6 6 6 6 6 6 6 6 7

7 7 7 7 7 7 7 7 7 7 7 7 7

Problem 10: An array is a container object that holds a fixed number of values of a single type. To create an array in C, we can do int arr[n];. Here, arr, is a variable array which holds up to  integers. The above array is a static array that has memory allocated at compile time. A dynamic array can be created in C, using the malloc function and the memory is allocated on the heap at runtime. To create an integer array,  of size , int \*arr = (int\*)malloc(n \* sizeof(int)), where  points to the base address of the array. When you have finished with the array, use free(arr) to deallocate the memory.

In this challenge, create an array of size  dynamically, and read the values from stdin. Iterate the array calculating the sum of all elements. Print the sum and free the memory where the array is stored.

While it is true that you can sum the elements as they are read, without first storing them to an array, but you will not get the experience working with an array. Efficiency will be required later.

**Input Format**

The first line contains an integer, .  
The next line contains  space-separated integers.

**Constraints**

**Output Format**

Print the sum of the integers in the array.

**Sample Input 0**

6

16 13 7 2 1 12

**Sample Output 0**

51

**Sample Input 1**

7

1 13 15 20 12 13 2

**Sample Output 1**

76

Problem 11: Given an array, of size , reverse it.

Example: If array, , after reversing it, the array should be, .

**Input Format**

The first line contains an integer, , denoting the size of the array. The next line contains  space-separated integers denoting the elements of the array.

**Constraints**

, where  is the  element of the array.

**Output Format**

The output is handled by the code given in the editor, which would print the array.

**Sample Input 0**

6

16 13 7 2 1 12

**Sample Output 0**

12 1 2 7 13 16

**Explanation 0**

Given array,  = . After reversing the array,  =

**Sample Input 1**

7

1 13 15 20 12 13 2

**Sample Output 1**

2 13 12 20 15 13 1

**Sample Input 2**

8

15 5 16 15 17 11 5 11

**Sample Output 2**

11 5 11 17 15 16 5 15

Problem 12: **Objective**  
This challenge will help you learn the concept of recursion.

A function that calls itself is known as a recursive function. The C programming language supports recursion. But while using recursion, one needs to be careful to define an exit condition from the function, otherwise it will go into an infinite loop.

To prevent infinite recursion,  statement (or similar approach) can be used where one branch makes the recursive call and other doesn't.

void recurse() {

.....

recurse() //recursive call

.....

}

int main() {

.....

recurse(); //function call

.....

}

**Task**

There is a series, , where the next term is the sum of pervious three terms. Given the first three terms of the series, , , and  respectively, you have to output the *nth* term of the series using recursion.

Recursive method for calculating *nth* term is given below.

**Input Format**

* The first line contains a single integer, .
* The next line contains *3* space-separated integers, , , and .

**Constraints**

**Output Format**

Print the *nth* term of the series, .

**Sample Input 0**

5

1 2 3

**Sample Output 0**

11

Problem 13: Given a string, , consisting of alphabets and digits, find the frequency of each digit in the given string.

**Input Format**

The first line contains a string,  which is the given number.

**Constraints**

All the elements of num are made of english alphabets and digits.

**Output Format**

Print ten space-separated integers in a single line denoting the frequency of each digit from  to .

**Sample Input 0**

a11472o5t6

**Sample Output 0**

0 2 1 0 1 1 1 1 0 0

**Explanation 0**

In the given string:

* occurs two times.
* and  occur one time each.
* The remaining digits  and  don't occur at all.

**Sample Input 1**

lw4n88j12n1

**Sample Output 1**

0 2 1 0 1 0 0 0 2 0

**Sample Input 2**

1v88886l256338ar0ekk

**Sample Output 2**

1 1 1 2 0 1 2 0 5 0

Problem 14: **Objective**

This challenge will help you to learn how to take a character, a string and a sentence as input in C.

To take a single character  as input, you can use scanf("%c", &ch ); and printf("%c", ch) writes a character specified by the argument char to stdout

char ch;

scanf("%c", &ch);

printf("%c", ch);

This piece of code prints the character .

You can take a string as input in C using scanf(“%s”, s). But, it accepts string only until it finds the first space.

In order to take a line as input, you can use scanf("%[^\n]%\*c", s); where  is defined as char s[MAX\_LEN] where  is the maximum size of . Here, [] is the scanset character. ^\n stands for taking input until a newline isn't encountered. Then, with this %\*c, it reads the newline character and here, the used \* indicates that this newline character is discarded.

**Note:** The statement: scanf("%[^\n]%\*c", s); will not work because the last statement will read a newline character, \n, from the previous line. This can be handled in a variety of ways. One way is to use scanf("\n"); before the last statement.

**Task**

You have to print the character, , in the first line. Then print  in next line. In the last line print the sentence, .

**Input Format**

First, take a character,  as input.  
Then take the string,  as input.  
Lastly, take the sentence  as input.

**Constraints**

Strings for  and  will have fewer than 100 characters, including the newline.

**Output Format**

Print three lines of output. The first line prints the character, .  
The second line prints the string, .  
The third line prints the sentence, .

**Sample Input 0**

C

Language

Welcome To C!!

**Sample Output 0**

C

Language

Welcome To C!!

Problem 15: Given a sentence, , print each word of the sentence in a new line.

**Input Format**

The first and only line contains a sentence, .

**Constraints**

**Output Format**

Print each word of the sentence in a new line.

**Sample Input 0**

This is C

**Sample Output 0**

This

is

C

**Explanation 0**

In the given string, there are three words ["This", "is", "C"]. We have to print each of these words in a new line.

**Sample Input 1**

Learning C is fun

**Sample Output 1**

Learning

C

is

fun

**Sample Input 2**

How is that

**Sample Output 2**

How

is

that

problem 16: You are transporting some boxes through a tunnel, where each box is a [parallelepiped](https://en.wikipedia.org/wiki/Parallelepiped), and is characterized by its length, width and height.

The height of the tunnel  feet and the width can be assumed to be infinite. A box can be carried through the tunnel only if its height is strictly less than the tunnel's height. Find the volume of each box that can be successfully transported to the other end of the tunnel. Note: Boxes cannot be rotated.

**Input Format**

The first line contains a single integer , denoting the number of boxes.  
 lines follow with three integers on each separated by single spaces  ,  and  which are length, width and height in feet of the -th box.

**Constraints**

**Output Format**

For every box from the input which has a height lesser than  feet, print its volume in a separate line.

**Sample Input 0**

4

5 5 5

1 2 40

10 5 41

7 2 42

**Sample Output 0**

125

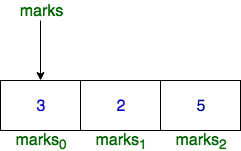
80

Problem 17: You are given an array of integers, , denoting the marks scored by students in a class.

* The alternating elements , ,  and so on denote the marks of boys.
* Similarly, , ,  and so on denote the marks of girls.

The array name, , works as a pointer which stores the base address of that array. In other words,  contains the address where  is stored in the memory.

For example, let  and  stores *0x7fff9575c05f*. Then, *0x7fff9575c05f* is the memory address of .



**Function Description**

Complete the function, *marks\_summation* in the editor below.

*marks\_summation* has the following parameters:

* *int marks[number\_of\_students]:* the marks for each student
* *int number\_of\_students:* the size of marks[]
* *char gender:* either 'g' or 'b'

**Returns**

* *int:* the sum of marks for boys if , or of marks of girls if

**Input Format**

* The first line contains , denoting the number of students in the class, hence the number of elements in .
* Each of the  subsequent lines contains .
* The next line contains .

**Constraints**

* (where )
* =  or

**Sample Input 0**

3

3

2

5

b

**Sample Output 0**

8

Problem 18: Snow Howler is the librarian at the central library of the city of HuskyLand. He must handle requests which come in the following forms:

*1 x y* : Insert a book with  pages at the end of the  shelf.

*2 x y* : Print the number of pages in the  book on the  shelf.

*3 x* : Print the number of books on the  shelf.

Snow Howler has got an assistant, Oshie, provided by the Department of Education. Although inexperienced, Oshie can handle all of the queries of types *2* and *3*.

Help Snow Howler deal with all the queries of type *1*.

Oshie has used two arrays:

int\* total\_number\_of\_books;

/\*

\* This stores the total number of books on each shelf.

\*/

int\*\* total\_number\_of\_pages;

/\*

\* This stores the total number of pages in each book of each shelf.

\* The rows represent the shelves and the columns represent the books.

\*/

**Input Format**

The first line contains an integer , the number of shelves in the library.  
The second line contains an integer , the number of requests.  
Each of the following  lines contains a request in one of the three specified formats.

**Constraints**

* For each query of the second type, it is guaranteed that a book is present on the  shelf at  index.
* Both the shelves and the books are numbered starting from 0.
* Maximum number of books per shelf .

**Output Format**

Write the logic for the requests of type 1. The logic for requests of types 2 and 3 are provided.

**Sample Input 0**

5

5

1 0 15

1 0 20

1 2 78

2 2 0

3 0

**Sample Output 0**

78

2

Problem 19: To sort a given array of strings into lexicographically increasing order or into an order in which the string with the lowest length appears first, a sorting function with a flag indicating the type of comparison strategy can be written. The disadvantage with doing so is having to rewrite the function for every new comparison strategy.

A better implementation would be to write a sorting function that accepts a pointer to the function that compares each pair of strings. Doing this will mean only passing a pointer to the sorting function with every new comparison strategy.

Given an array of strings, you need to implement a  function which sorts the strings according to a comparison function, i.e, you need to implement the function :

void string\_sort(const char \*\*arr,const int cnt, int (\*cmp\_func)(const char\* a, const char\* b)){

}

The arguments passed to this function are:

* an array of strings :
* length of string array:
* pointer to the string comparison function:

You also need to implement the following four string comparison functions:

1. to sort the strings in lexicographically non-decreasing order.
2. to sort the strings in lexicographically non-increasing order.
3. to sort the strings in non-decreasing order of the number of distinct characters present in them. If two strings have the same number of distinct characters present in them, then the lexicographically smaller string should appear first.
4. to sort the strings in non-decreasing order of their lengths. If two strings have the same length, then the lexicographically smaller string should appear first.

**Input Format**

You just need to complete the function string\\_sort and implement the four string comparison functions.

**Constraints**

* No. of Strings
* Total Length of all the strings
* You have to write your own sorting function and you cannot use the inbuilt  function
* The strings consists of lower-case English Alphabets only.

**Output Format**

The locked code-stub will check the logic of your code. The output consists of the strings sorted according to the four comparsion functions in the order mentioned in the problem statement.

**Sample Input 0**

4

wkue

qoi

sbv

fekls

**Sample Output 0**

fekls

qoi

sbv

wkue

wkue

sbv

qoi

fekls

qoi

sbv

wkue

fekls

qoi

sbv

wkue

fekls

problem 20: Strings are usually ordered in lexicographical order. That means they are ordered by comparing their leftmost different characters. For example,  because . Also  because . If one string is an exact prefix of the other it is lexicographically smaller, e.g., .

Given an array of strings sorted in lexicographical order, print all of its permutations in strict lexicographical order. If two permutations look the same, only print one of them. See the 'note' below for an example.

Complete the function next\_permutation which generates the permutations in the described order.

For example, . The six permutations in correct order are:

ab bc cd

ab cd bc

bc ab cd

bc cd ab

cd ab bc

cd bc ab

**Note:** There may be two or more of the same string as elements of .  
For example, . Only one instance of a permutation where all elements match should be printed. In other words, if , then print either  or  but not both.

A three element array having three distinct elements has six permutations as shown above. In this case, there are three matching pairs of permutations where  and  are switched. We only print the three visibly unique permutations:

ab ab bc

ab bc ab

bc ab ab

**Input Format**

The first line of each test file contains a single integer , the length of the string array .

Each of the next  lines contains a string .

**Constraints**

* contains only lowercase English letters.

**Output Format**

Print each permutation as a list of space-separated strings on a single line.

**Sample Input 0**

2

ab

cd

**Sample Output 0**

ab cd

cd ab

**Sample Input 1**

3

a

bc

bc

**Sample Output 1**

a bc bc

bc a bc

bc bc a

problem 21: Variadic functions are functions which take a variable number of arguments. In C programming, a variadic function will contribute to the flexibility of the program that you are developing.

The declaration of a variadic function starts with the declaration of at least one named variable, and uses an ellipsis as the last parameter, e.g.

int printf(const char\* format, ...);

In this problem, you will implement three variadic functions named ,  and  to calculate sums, minima, maxima of a variable number of arguments. The first argument passed to the variadic function is the count of the number of arguments, which is followed by the arguments themselves.

**Input Format**

* The first line of the input consists of an integer .
* Each test case tests the logic of your code by sending a test implementation of 3, 5 and 10 elements respectively.
* You can test your code against sample/custom input.
* The error log prints the parameters which are passed to the test implementation. It also prints the sum, minimum element and maximum element corresponding to your code.

**Constraints**

 .

**Output Format**

"Correct Answer" is printed corresponding to each correct execution of a test implementation."Wrong Answer" is printed otherwise.

**Sample Input 0**

1

**Sample Output 0**

Correct Answer

Correct Answer

Correct Answer

Problem 22: A document is represented as a collection paragraphs, a paragraph is represented as a collection of sentences, a sentence is represented as a collection of words and a word is represented as a collection of lower-case ([a-z]) and upper-case ([A-Z]) English characters.

You will convert a raw text document into its component paragraphs, sentences and words. To test your results, queries will ask you to return a specific paragraph, sentence or word as described below.

Alicia is studying the C programming language at the University of Dunkirk and she represents the words, sentences, paragraphs, and documents using pointers:

* A word is described by .
* A sentence is described by . The words in the sentence are separated by one space (" "). The last word does not end with a space(" ").
* A paragraph is described by . The sentences in the paragraph are separated by one period (".").
* A document is described by . The paragraphs in the document are separated by one newline("\n"). The last paragraph does not end with a newline.

For example:

Learning C is fun.  
Learning pointers is more fun.It is good to have pointers.

* The only sentence in the first paragraph could be represented as:

char\*\* first\_sentence\_in\_first\_paragraph = {"Learning", "C", "is", "fun"};

* The first paragraph itself could be represented as:

char\*\*\* first\_paragraph = {{"Learning", "C", "is", "fun"}};

* The first sentence in the second paragraph could be represented as:

char\*\* first\_sentence\_in\_second\_paragraph = {"Learning", "pointers", "is", "more", "fun"};

* The second sentence in the second paragraph could be represented as:

char\*\* second\_sentence\_in\_second\_paragraph = {"It", "is", "good", "to", "have", "pointers"};

* The second paragraph could be represented as:

char\*\*\* second\_paragraph = {{"Learning", "pointers", "is", "more", "fun"}, {"It", "is", "good", "to", "have", "pointers"}};

* Finally, the document could be represented as:

char\*\*\*\* document = {{{"Learning", "C", "is", "fun"}}, {{"Learning", "pointers", "is", "more", "fun"}, {"It", "is", "good", "to", "have", "pointers"}}};

Alicia has sent a document to her friend Teodora as a string of characters, i.e. represented by  not . Help her convert the document to  form by completing the following functions:

* to return the document represented by .
* to return the  paragraph.
* to return the  sentence in the  paragraph.
* to return the  word in the  sentence of the  paragraph.

**Input Format**

The first line contains the integer .  
Each of the next  lines contains a paragraph as a single string.  
The next line contains the integer , the number of queries.  
Each of the next  lines or groups of lines contains a query in one of the following formats:

* 1 The first line contains :
  + The next line contains an integer , the number of sentences in the  paragraph.
  + Each of the next  lines contains an integer , the number of words in the  sentence.
  + This query corresponds to calling the function .
* 2 The first line contains :
  + The next line contains an integer , the number of words in the  sentence of the  paragraph.
  + This query corresponds to calling the function
* 3 The only line contains :
  + This query corresponds to calling the function

**Constraints**

* The text which is passed to the  has words separated by a space (" "), sentences separated by a period (".") and paragraphs separated by a newline("\n").
* The last word in a sentence does not end with a space.
* The last paragraph does not end with a newline.
* The words contain only upper-case and lower-case English letters.
* number of characters in the entire document
* number of paragraphs in the entire document

**Output Format**

Print the paragraph, sentence or the word corresponding to the query to check the logic of your code.

**Sample Input 0**

2

Learning C is fun.

Learning pointers is more fun.It is good to have pointers.

3

1 2

2

5

6

2 1 1

4

3 1 1 1

**Sample Output 0**

Learning pointers is more fun.It is good to have pointers.

Learning C is fun

Learning

Problem 23: You are given  triangles, specifically, their sides ,  and . Print them in the same style but sorted by their areas from the smallest one to the largest one. It is guaranteed that all the areas are different.

The best way to calculate a area of a triangle with sides ,  and  is Heron's formula:

 where .

**Input Format**

The first line of each test file contains a single integer .  lines follow with three space-separated integers, ,  and .

**Constraints**

* , and

**Output Format**

Print exactly  lines. On each line print  space-separated integers, the ,  and  of the corresponding triangle.

**Sample Input 0**

3

7 24 25

5 12 13

3 4 5

**Sample Output 0**

3 4 5

5 12 13

7 24 25

**Explanation 0**

The square of the first triangle is . The square of the second triangle is . The square of the third triangle is . So the sorted order is the reverse one.

Change Theme

Language: C

More

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

            }

        }

}

}

int main()

{

    int n;

    scanf("%d", &n);

    triangle \*tr = malloc(n \* sizeof(triangle));

    for (int i = 0; i < n; i++) {

        scanf("%d%d%d", &tr[i].a, &tr[i].b, &tr[i].c);

    }

    sort\_by\_area(tr, n);

    for (int i = 0; i < n; i++) {

        printf("%d %d %d\n", tr[i].a, tr[i].b, tr[i].c);

    }

    return 0;

}



Line: 48 Col: 2

Submit Code

Run Code

Upload Code as File



**Test against custom input**

C language

You have earned 30.00 points!

23/25 challenges solved.

**92%**

**Test case 0**

Test case 1

Test case

Test case 3

Test case 4

Test case 5

Test case 6

Test case 7

Test case 8

Test case 9

Test case 10

Compiler Message

Success

Input (stdin)

Download

* **3**
* **7 24 25**
* **5 12 13**
* **3 4 5**

Expected Output

Download

* **3 4 5**
* **5 12 13**
* **7 24 25**
* [Blog](https://www.hackerrank.com/blog)

Problem 24:

We live in a big country. This country has  towns in it. Each town has some post offices in which packages are stored and transferred.

Post offices have different inner structure. Specifically, each of them has some limitations on the packages it can store - their weight should be between  and  inclusively, where  and  are fixed for each office.

Packages are stored in some order in the office queue. That means, that they are processed using this order when sending and receiving.

Sometimes two post offices, even in different towns, may organize the following transaction: the first one sends all its packages to the second one. The second one accepts the packages that satisfy the weight condition for the second office and rejects all other ones. These rejected packages return to the first office back and are stored in the same order they were stored before they were sent. The accepted packages move to the tail of the second office's queue in the same order they were stored in the first office.

You should process several queries in your program. You'll be provided with structures ,  and . in order to complete this task, you should fill the following functions:

 - given the town , print all packages in this town. They should be printed as follows:

Town\_name:

0:

id\_0

id\_1

...

1:

id\_2

id\_3

...

...

where ,  etc are the numbers of post offices and , ... are the ids of packages from the th post office in the order of its queue, ,  are from the st one etc. There should be one '\t' symbol before post office numbers and two '\t' symbols before the ids.

 - given the towns  and  and post office indices  and , manage the transaction described above between the post office # in town  and the post office # in town .

 - given all towns, find the one with the most number of packages in all post offices altogether. If there are several of them, find the first one from the collection .

 - given all towns and a string , find the town with the name . It's guaranteed that the town exists.

**Input Format**

First line of the input contains a single integer .  blocks follow, each describing a town.

Every town block contains several lines. On the first line there is a string  - the name of the town. On the second line there is an integer  - the number of the offices in the town.  blocks follow then, each describing an office.

Every office block also contains several lines. On the first line there are three integers separated by single spaces:  (the number of packages in the office),  and  (described above).  blocks follow, each describing a package.

Every package block contains exactly two lines. On the first line there is a string  which is an id of the package. On the second line there is an integer  which is the weight of the package.

Then, there is a single integer  on the line which is the number of queries.  blocks follow, each describing a query.

Every query block contains several lines. On the first line there is an integer ,  or . If this integer is , on the second line there is a string  - the name of town for which all packages should be printed. If this integer is , on the second line there are string , integer , string  and integer  separated by single spaces. That means transactions between post office # in the town  and post office # in the town  should be processed.

If the integer is , no lines follow and the town with the most number of packages should be found.

**Constraints**

* All integer are between  and
* , .
* All strings have length
* All towns' names have only uppercase english letters and are unique.
* All packages' ids have only lowercase english letters and are unique.
* For each post office,   .
* All queries are valid, that means, towns with the given names always exist, post offices with the given indices always exist.

**Output Format**

For queries of type , print all packages in the format provided in the problem statement. For queries of type , print "Town with the most number of packages is " on a separate line.

**Sample Input 0**

2

A

2

2 1 3

a 2

b 3

1 2 4

c 2

B

1

4 1 4

d 1

e 2

f 3

h 4

5

3

2 B 0 A 1

3

1 A

1 B

**Sample Output 0**

Town with the most number of packages is B

Town with the most number of packages is A

A:

0:

a

Problem 25: A document is represented as a collection paragraphs, a paragraph is represented as a collection of sentences, a sentence is represented as a collection of words and a word is represented as a collection of lower-case ([a-z]) and upper-case ([A-Z]) English characters. You will convert a raw text document into its component paragraphs, sentences and words. To test your results, queries will ask you to return a specific paragraph, sentence or word as described below.

Alicia is studying the C programming language at the University of Dunkirk and she represents the words, sentences, paragraphs, and documents using pointers:

* A word is described by:

struct word {

char\* data;

};

* A sentence is described by:

struct sentence {

struct word\* data;

int word\_count;//the number of words in a sentence

};

The words in the sentence are separated by one space (" "). The last word does not end with a space.

* A paragraph is described by:

struct paragraph {

struct sentence\* data ;

int sentence\_count;//the number of sentences in a paragraph

};

The sentences in the paragraph are separated by one period (".").

* A document is described by:

struct document {

struct paragraph\* data;

int paragraph\_count;//the number of paragraphs in a document

};

The paragraphs in the document are separated by one newline("\n"). The last paragraph does not end with a newline.

For example:

Learning C is fun.  
Learning pointers is more fun.It is good to have pointers.

* The only sentence in the first paragraph could be represented as:

struct sentence first\_sentence\_in\_first\_paragraph;

first\_sentence\_in\_first\_paragraph.data = {"Learning", "C", "is", "fun"};

* The first paragraph itself could be represented as:

struct paragraph first\_paragraph;

**Hackerearth**

**Problem : 26-50**

**Gmail:** [moinulislamsajid107@gmail.com](mailto:moinulislamsajid107@gmail.com)

Password : Sajid\_20



Basic Programming

Problem 26:

Zoo

You are required to enter a word that consists of � and � that denote the number of **Z**s and **O**s respectively. The input word is considered similar to word **zoo**if 2×�=�.

Determine if the entered word is similar to word **zoo**.

For example, words such as **zzoooo**and **zzzoooooo**are similar to word **zoo**but not the words such as **zzooo** and **zzzooooo**.

**Input format**

* First line: A word that starts with several **Z**s and continues by several **O**s.  
  **Note**: The maximum length of this word must be 20.

**Output format**

Print **Yes**if the input word can be considered as the string **zoo**otherwise, print **No**.

**Sample Input**

zzzoooooo

**Sample Output**

Yes

Problem 27:

**Divisibility**

You are provided an array � of size � that contains non-negative integers. Your task is to determine whether the number that is formed by selecting the last digit of all the N numbers is divisible by 10.

**Note**: View the sample explanation section for more clarification.

**Input format**

* First line: A single integer � denoting the size of array �
* Second line: � space-separated integers.

**Output format**

If the number is divisible by 10, then print ���. Otherwise, print ��.

**Constraints**1≤�≤1050≤�[�]≤105

**Sample Input**

5

85 25 65 21 84

**Sample Output**

No

Problem 28:

**Split houses**

You live in a village. The village can be represented as a line that contains � grids. Each grid can be denoted as a house that is marked as � or a blank space that is marked as ..

A person lives in each house. A person can move to a grid if it is adjacent to that person. Therefore, the grid must be present on the left and right side of that person.

Now, you are required to put some fences that can be marked as � on some blank spaces so that the village can be divided into several pieces. A person cannot walk past a fence but can walk through a house.

You are required to divide the house based on the following rules:

* A person cannot reach a house that does not belong to that specific person.
* The number of grids each person can reach is the same and it includes the grid in which the house is situated.
* In order to show that you are enthusiastic and if there are many answers, then you are required to print the one where most fences are placed.

Your task is to decide whether there is a possible solution. Print the possible solution.

**Input format**

* First line: An integer � that represents the number of grids (1≤�≤20)
* Second line: � characters that indicate the villages that are represented as � or .

**Output format**

The output must be printed in the following format:

* First line: If possible, then print ���. Otherwise, print ��.
* Second line: If the answer is ���, then print the way to do so.

**Sample Input**

5

H...H

**Sample Output**

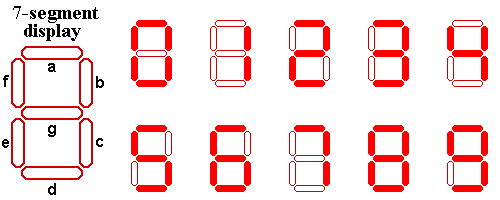
YES

HBBBH

Problem29:

7 segment display

You all must have seen a seven segment display.If not here it is:



Alice got a number written in **seven segment format** where each segment was creatted used a matchstick.

Example: If Alice gets a number **123**so basically Alice used **12 matchsticks**for this number.

Alice is wondering what is the numerically largest value that she can generate by using **at most** the matchsticks that she currently possess.Help Alice out by telling her that number.

**Input Format:**

First line contains **T**(test cases).

Next **T**lines contain a Number **N**.

**Output Format:**

Print the **largest**possible number **numerically**that can be generated using **at max** that many number of matchsticks which was used to generate N.

**Constraints:**

1≤�≤100

1≤�����ℎ(�)≤100

**Sample Input**

2

1

0

**Sample Output**

1

111

Problem 30:

**Ali and Helping innocent people**

Arpasland has surrounded by attackers. A truck enters the city. The driver claims the load is food and medicine from Iranians. Ali is one of the soldiers in Arpasland. He doubts about the truck, maybe it's from the siege. He knows that a tag is valid if the sum of every two consecutive digits of it is even and its letter is not a vowel. Determine if the tag of the truck is valid or not.

We consider the letters **"A","E","I","O","U","Y**" to be vowels for this problem.

**Input Format**

The first line contains a string of length 9. The format is "DDXDDD-DD", where D stands for a digit (non zero) and X is an uppercase english letter.

**Output Format**

Print "valid" (without quotes) if the tag is valid, print "invalid" otherwise (without quotes)

**Sample Input**

12X345-67

**Sample Output**

invalid

Problem 31:

TWO Steps of numbers

You are given an array � of � elements. Now you need to choose the best index of this array �. An index of the array is called best if the special sum of this index is maximum across the special sum of all the other indices.

To calculate the special sum for any index � , you pick the first element that is �[�] and add it to your sum. Now you pick next two elements i.e. �[�+1] and �[�+2] and add both of them to your sum. Now you will pick the next 3 elements and this continues till the index for which it is possible to pick the elements. For example:

If our array contains 10 elements and you choose index to be 3 then your special sum is denoted by -  
(�[3])+(�[4]+�[5])+(�[6]+�[7]+�[8]) , beyond this its not possible to add further elements as the index value will cross the value 10.

Find the best index and in the output print its corresponding special sum. Note that there may be more than one best indices but you need to only print the maximum special sum.

**Input**  
First line contains an integer � as input. Next line contains � space separated integers denoting the elements of the array �.  
**Output**  
In the output you have to print an integer that denotes the maximum special sum

**Constraints**  
1≤�≤105  
−107≤�[�]≤107

|  |  |
| --- | --- |
| **Sample Inputs** | |
| **Input** | **Output** |
| 5 1 3 1 2 5 | 8 |
| 10 2 1 3 9 2 4 -10 -9 1 3 | 9 |

**Sample Input**

6

-3 2 3 -4 3 1

**Sample Output**

3

Problem 32:

**Best Index**

You are given two arrays �1,�2,…,�� and �1,�2,…,��. In each step, you can set ��=��−�� if ��≥��. Determine the minimum number of steps that are required to make all �'s equal.

**Input format**

* First line: �
* Second line: �1,�2,…,��
* Third line: �1,�2,…,��

**Output format**

Print the minimum number of steps that are required to make all �'s equal. If it is not possible, then print **-1**.

**Constraints**

1≤�, ��, ��≤5000

**Sample input**

2  
5 6  
4 3

**Sample output**

-1

**Sample Input**

5

5 7 10 5 15

2 2 1 3 5

**Sample Output**

8

Problem 33:

**Maximum borders**

You are given a table with � rows and � columns. Each cell is colored with white or black. Considering the shapes created by black cells, what is the maximum border of these shapes? Border of a shape means the maximum number of consecutive black cells in any row or column without any white cell in between.

A shape is a set of connected cells. Two cells are connected if they share an edge. Note that no shape has a hole in it.

**Input format**

* The first line contains � denoting the number of test cases.
* The first line of each test case contains integers �, � denoting the number of rows and columns of the matrix. Here, '**#**' represents a black cell and '**.**' represents a white cell.
* Each of the next � lines contains� integers.

**Output format**

Print the maximum border of the shapes.

**Sample Input**

10

2 15

.....####......

.....#.........

7 9

...###...

...###...

..#......

.####....

..#......

...#####.

.........

18 11

.#########.

########...

.........#.

####.......

.....#####.

.....##....

....#####..

.....####..

..###......

......#....

....#####..

...####....

##.........

#####......

....#####..

....##.....

.#######...

.#.........

1 15

.....######....

5 11

..#####....

.#######...

......#....

....#####..

...#####...

8 13

.....######..

......##.....

########.....

...#.........

.............

#######......

..######.....

####.........

7 5

.....

..##.

###..

..##.

.....

..#..

.#...

14 2

..

#.

..

#.

..

#.

..

..

#.

..

..

..

#.

..

7 15

.###########...

##############.

...####........

...##########..

.......#.......

.....#########.

.#######.......

12 6

#####.

###...

#.....

##....

###...

......

.##...

..##..

...#..

..#...

#####.

####..

Problem 34

**Cost of balloons**

You are conducting a contest at your college. This contest consists of two problems and � participants. You know the problem that a candidate will solve during the contest.

You provide a balloon to a participant after he or she solves a problem. There are only green and purple-colored balloons available in a market. Each problem must have a balloon associated with it as a prize for solving that specific problem. You can distribute balloons to each participant by performing the following operation:

1. Use green-colored balloons for the first problem and purple-colored balloons for the second problem
2. Use purple-colored balloons for the first problem and green-colored balloons for the second problem

You are given the cost of each balloon and problems that each participant solve. Your task is to print the minimum price that you have to pay while purchasing balloons.

**Input format**

* First line: � that denotes the number of test cases (1≤�≤10)
* For each test case:
  + First line: Cost of green and purple-colored balloons
  + Second line: � that denotes the number of participants (1≤�≤10)
* Next � lines: Contain the status of users. For example, if the value of the ��ℎ integer in the ��ℎ row is 0, then it depicts that the ��ℎ participant has not solved the ��ℎ problem. Similarly, if the value of the ��ℎ integer in the ��ℎ row is 1, then it depicts that the ��ℎ participant has solved the ��ℎ problem.

**Output format**  
For each test case, print the minimum cost that you have to pay to purchase balloons.

**Sample Input**

2

9 6

10

1 1

1 1

0 1

0 0

0 1

0 0

0 1

0 1

1 1

0 0

1 9

10

0 1

0 0

0 0

0 1

1 0

0 1

0 1

0 0

0 1

0 0

Problem 35:

**Toggle String**

You have been given a String *S* consisting of uppercase and lowercase English alphabets. You need to change the case of each alphabet in this String. That is, all the uppercase letters should be converted to lowercase and all the lowercase letters should be converted to uppercase. You need to then print the resultant String to output.

**Input Format**  
The first and only line of input contains the String *S*

**Output Format**  
Print the resultant String on a single line.

**Constraints**  
1≤|�|≤100 *where S denotes the length of string S*.

**Sample Input**

abcdE

**Sample Output**

ABCDe

Probem 36:

**Find Product**

You have been given an array *A* of size *N* consisting of positive integers. You need to find and print the product of all the number in this array **Modulo** 109+7.

**Input Format**:  
The first line contains a single integer *N* denoting the size of the array. The next line contains *N* space separated integers denoting the elements of the array

**Output Format**:  
Print a single integer denoting the product of all the elements of the array Modulo 109+7.

**Constraints**:  
1≤�≤103  
1≤�[�]≤103

**Sample Input**

5

1 2 3 4 5

**Sample Output**

120

Problem 37:

**Palindromic String**

You have been given a String *S*. You need to find and print whether this string is a palindrome or not. If yes, print "YES" (without quotes), else print "NO" (without quotes).

**Input Format**  
The first and only line of input contains the String *S*. The String shall consist of lowercase English alphabets only.

**Output Format**  
Print the required answer on a single line.

**Constraints** 1≤|�|≤100

**Note**  
String *S* consists of lowercase English Alphabets only.

**Sample Input**

aba

**Sample Output**

YES

Problem 38:

**Life, the Universe, and Everything**

Your program is to use the brute-force approach in order to find the Answer to Life, the Universe, and Everything. More precisely... rewrite small numbers from input to output. Stop processing input after reading in the number 42. All numbers at input are integers of one or two digits.

**Sample Input**

1

2

88

42

99

**Sample Output**

1

2

88

Problem 39:

**Count Divisors**

You have been given 3 integers - l, r and k. Find how many numbers between l and r (both inclusive) are divisible by k. You do not need to print these numbers, you just have to find their count.

**Input Format**  
The first and only line of input contains 3 space separated integers *l*, *r* and *k*.

**Output Format**  
Print the required answer on a single line.

**Constraints**  
1≤�≤�≤1000  
1≤�≤1000

**Sample Input**

1 10 1

**Sample Output**

10

Problem 40:

**Factorial**

You have been given a positive integer N. You need to find and print the **Factorial** of this number. The Factorial of a positive integer N refers to the product of all number in the range from 1 to N. You can read more about the factorial of a number [here.](https://en.wikipedia.org/wiki/Factorial)

**Input Format**:  
The first and only line of the input contains a single integer N denoting the number whose factorial you need to find.

**Output Format**  
Output a single line denoting the factorial of the number N.

**Constraints**  
1≤�≤10

**Sample Input**

2

**Sample Output**

2

Time Limit: 1

Problem 41:

**Roy and Profile Picture**

Roy wants to change his profile picture on Facebook. Now Facebook has some restriction over the dimension of picture that we can upload.  
Minimum dimension of the picture can be **L x L**, where **L** is the length of the side of square.

Now Roy has **N** photos of various dimensions.  
Dimension of a photo is denoted as **W x H**  
where **W** - width of the photo and **H** - Height of the photo

When any photo is uploaded following events may occur:

[1] If any of the width or height is less than L, user is prompted to upload another one. Print "**UPLOAD ANOTHER**" in this case.  
[2] If width and height, both are large enough and  
(a) if the photo is already square then it is accepted. Print "**ACCEPTED**" in this case.  
(b) else user is prompted to crop it. Print "**CROP IT**" in this case.

(quotes are only for clarification)

Given L, N, W and H as input, print appropriate text as output.

**Input:**  
First line contains **L**.  
Second line contains **N**, number of photos.  
Following N lines each contains two space separated integers **W** and **H**.

**Output:**  
Print appropriate text for each photo in a new line.

Problem42:

**Going to office**

Alice has the following two types of taxis:

* Online taxi: It can be booked by using an online application from phones
* Classic taxi: It can be booked anywhere on the road

The online taxis cost �� for the first �� km and �� for every km afterward. The classic taxis travel at a speed of �� km per minute. The cost of classic taxis are ��, ��, and ��that represents the base fare, costfor every minute that is spent in the taxi, and cost for each kilometer that you ride.

You are going to the office from your home. Your task is to minimize the cost that you are required to pay. The distance from your home to the office is D. You are required to select whether you want to use online or classic taxis to go to your office. If both the taxis cost the same, then you must use an online taxi.

**Input format**

* First line: Single integer � that denotes the distance from your house to the office
* Next line: Three integers ��*,*��, and��
* Next line: Four integers ��, ��, ��, and ��

**Output format**

If you select an online taxi to travel, then print a string '**Online Taxi**'. Otherwise, select '**Classic Taxi**'. You can print this string in a new, single line.

**Constraints**

1≤�, ��, ��, ��, ��, ��, ��, �� ≤ 109

**Sample Input**

13

6 7 4

4 2 1 2

**Sample Output**

Online Taxi

Problem43:

**Let Us Understand Computer**

Mr. ABC was recently learning about computer division. Considering the basic model of the computer suppose we wish to divide a number **X** by **D** i.e **X/D** and obtain the result (Note that it is integer division i.e result of 7/2 will be 3).

Now the computer will give the divide overflow error if:

The number of bits in the binary representation(without appending any leading zeroes) of the resulting number(quotient) is greater than the number of bits in the binary representation of divisor(**D**) (Without appending any leading zeroes).

Now Mr. ABC is given an integer **X** for **T** testcases .He wishes to find number of such divisors **D(1<=D<=X)** for which divide overflow will **not** occur.

***CONSTRAINTS*** :

1≤�≤106

1≤�≤1012

***INPUT:***

The first line contains the integer **T ,**the number of test cases.

In next **T**lines, each line contains an integer **X.**

***OUTPUT:***

Print the required output for each testcase in new line.

**Sample Input**

2

3

5

**Sample Output**

2

4

Problem44:

**Robotic moves**

Problem 45:

**Minimize Cost**

You are given an array of numbers �� which contains positive as well as negative numbers . The cost of the array can be defined as �(�)

�(�)=|�1+�1|+|�2+�2|+........|��+��| , where T is the transfer array which contains *N* zeros initially.

You need to minimize this cost . You can transfer value from one array element to another if and only if the distance between them is at most *K*.

Also, transfer value can't be transferred further.

Say array contains 3,−1,−2 and �=1

if we transfer 3 from 1�� element to 2�� , the array becomes

Original Value 3,−1,−2

Transferred value −3,3,0

�(�)=|3−3|+|−1+3|+........|−2+0|=4 which is minimum in this case

**Note :**

**Only positive value can be transferred**

**It is not necessary to transfer whole value i.e partial transfer is also acceptable. This means that if you have**�[�]=5**then you can distribute the value 5 across many other array elements provided that they finally sum to a number less than equal to 5. For example 5 can be transferred in chunks of smaller values say 2 , 3 but their sum should not exceed 5.**

**Input:**

First line contains *N* and *K* separated by space

Second line denotes an array of size *N*

**Output**

Minimum value of �(�)

**Constraints**

1≤�,�≤105

−109≤��≤109

**Sample Input**

3 2

3 -1 -2

**Sample Output**

0

**Problem 46:**

**GCD Strings**

**Problem 47:**

**N-Queens**

**Problem 48:**

**Excursion**

**Problem 49:**

**Erasing an array**

**Problem 50:**

**Simon cannot sleep**