1. You are a shoplifter and you have broke into a shop. You see N items that you can take and you know the price for every item. The problem is that you can only take K items. Write a program that will output the maximum amount you can steal.

Input: In the first line input the two numbers, N and K. In the following N lines input the price of each item.

Output: In a single line print the result.

```
Input:
5 2
4
6
3
9
2

Output:

15
```

2. Given an array **arr[]** of **N** non-negative integers and an integer **K**, the idea is to find the length of the longest subsequence having Xor of adjacent elements equal to **K**.

Examples:

Input: *N = 5, arr[] = {3, 2, 4, 3, 5}, K = 1*
Output: *3*
Explanation:
All the subsequences having Xor of adjacent element equal to K are {3, 2}, {2, 3}, {4, 5}, {3, 2, 3}.
Therefore, the length of the longest subsequence having xor of adjacent element as 1 is 3.

Input: *N = 8, arr[] = {4, 5, 4, 7, 3, 5, 4, 6}, K = 2* ***Output:*** *3*
Explanation:
All the subsequences having Xor of adjacent element equal to K are {4, 6}, {5, 7}, {7, 5}, {5, 7, 5}.Therefore, the length of the longest subsequence having xor of adjacent element as 1 is 3

3. An intern at HackerRank is assigned to finding the optimal middle subsequence. An optimal middle subsequence is the subsequence chosen of length 3 chosen from an array arr, such that chosen[0] < chosen[1] > chosen[2] and that the sum of its elements is the minimum possible. Given an array, return the sum of the values of the optimal middle subsequence. If there is none, return -1. Write C++ code.

4. Given an array of N elements and each element is either 1 or 0. You need to make all the elements of the array equal to 0 by performing the below operations:

- If an element is 1, You can change it's value equal to 0 then,
    - if the next consecutive element is 1, it will automatically get converted to 0.
    - if the next consecutive element is already 0, nothing will happen.

Now, the task is to find the minimum number of operations required to make all elements equal to 0.

Examples:

```
Input : arr[] = {1, 1, 0, 0, 1, 1, 1, 0, 0, 1}
Output : Minimum changes: 3
```

```
Input : arr[] = {1, 1, 1, 1}
Output : Minimum changes: 1
```

5. CORRECT THE CODE SNIPPET

How can the issue in the following code be solved ? Select all that apply

```
int* add_numbers(int, int);
Void main()
{
int* p;
p=add number(1,3);
}
int* add_numbers(int a, int b)
{
int* sum = (int*)malloc(16);
*sum = a+b;
return sum;
}
```

6.**Pointer Arithmetic**
What is the output of the following code snippet ?

```
#include <stdio.h>
Void main()
{
Int x=4;
Int *p =&x;
Int *k =p++;
Int r = p-k;
Print f(“%d”,r);
}
```

Pick one option

- ☐ 4
- ☐ 2
- ☐ 1
- ☐ 5

7. Dynamic memory allocation

With respect to the differences between malloc and calloc, select the option(s) that are true

Pick ONE OR MORE options

a) Calloc initialises the allocated memory with 0 (zero) while malloc just fills it with garbage
b) Calloc always allocates contiguous memory locations but malloc may or may not allocate contiguous memory locations
c) Calloc takes the number and type of data as arguments while malloc takes the number of bytes
d) Calloc may or may not allocate contiguous memory locations but malloc always allocates contiguous memory locations

8.output of the following code is :

```
 #include <stdio.h>

void main()
{
```

```
Int a[ ] = {1,2,4,6,8};
Int *p[ ] = {a, a+1, a+2, a+3, a+4);
int** p1=p;
int* p2 = *(p+2);
printf( “%u %u %u\n” , *++p2, ++*p2++ , ++++++p1);

}
```

9.Which of the following is the correct declaration for constant pointer to an int datatype?

Pick one option :

a) const int *p;
b) int* const p;
c) Int const *p;
d) Int const *const p;

10. Given that 1 char = 1 byte, 1 int = 2 bytes and 1 float = 4 bytes. What is the output of the following code :

```
#include <stdio.h>
int main (void)
{
Char *a = (char *)malloc(12);
printf(“%d”, sizeof(a));
printf(“%d\n”, sizeof(*a));
return 0;
}
```

Pick one option

a) 4 1
b) 12 1
c) 1 1
d) 2 1

11. Predict the output of the following program :

```
#include <stdio.h>
int main ( )
{
        int i = 109;
```

```
if(i>50)

i++;
i+=2;

else

i-- ;

printf("%d\n\n\n",i);
```

12. Consider the following code

```
        #include <stdio.h>
                int main ( )
                {
                        char i=101 ;
                while(--i)
                {
                    printf("\Hello World");
                        i- - ;
                }
                        return 0;
                }
```

Given that ASCII value of ‘0’ is 48;
How many times the string “Hello World” will be printed:

Pick one option
a)101
b)50
c)Infinitely many
d)51
e)26