# GUESS THE OUTPUT ANSWERS

## ROUND-1

QUESTION - 1:

```
x = {0} < {1}
print(x)
```

----- ( OUTPUT: False ) -----

QUESTION - 2:

```
print("_ : " + ("Underscore" * (-1)))
```

----- ( OUTPUT: _ : ) -----

QUESTION - 3:

```
#include <stdio.h>
void solve()
{
   int ch = 2;
   switch (ch)
   {
   case 1:
      printf("1 ");

   case 2:
      printf("2 ");

   case 3:
      printf("3 ");

   default:
      printf("None ");
   }
}
```

```
int main()
{
   solve();
   return 0;
}
```

----- ( OUTPUT: 2 3 None ) -----

QUESTION - 4:

```
x = "banana"
y = x.rstrip("an")
print(y)
```

----- ( OUTPUT: b ) -----

QUESTION - 5:

```
#include <stdio.h>
int main()
{
   int a[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
   int sum = 0;
   for (int i = 2; i < 6; i++)
   {
      sum += a[i];
   }
   printf("%d", sum);
   return 0;
}
```

----- ( OUTPUT: 18 ) -----

QUESTION - 6:

```
#include <stdio.h>
void solve()
{
   printf("%d %d", (023), (23));
```

```
}
int main()
{
   solve();
   return 0;
}
```

----- ( OUTPUT: 19 23 ) -----

QUESTION - 7:

```
case1 = 1
case2 = 2

case1 = case1^case2
case2 = case1^case2
case1 = case1^case2

print(case1)
print(case2)
```

----- ( OUTPUT: 2
1 ) -----

QUESTION - 8:

```
x = {1, 2, 2, 3}
y = x.discard(2)
z = sum(x)
print(z)
```

----- ( OUTPUT: 4 ) -----

QUESTION - 9:

```
x = list(range(-3))
y = x * 2
z = y[0] + 1
print(z)
```

----- ( OUTPUT: IndexError: List index out of range ) -----

QUESTION - 10:

```
#include<stdio.h>
int main()
{
   int x = 10, y = 3, z = 2;
   int ans = ++x + x++ - --y + z++ * ++y;
   printf("%d\n", ans);
   return 0;
}
```

----- ( OUTPUT: 26 ) -----

QUESTION - 11:

```
List = ["P", 20, "R", 10, "S", 30]
Times = 0
Alpha = ""
Sum = 0
for I in range(1,6,2):
   Times = Times + I
   Alpha = Alpha + List[I -1] + "#"
   Sum = Sum + List[I]
print(Times, Sum, Alpha)
```

----- ( OUTPUT: 9 60 P#R#S# ) -----

QUESTION - 12:

```
#include <stdio.h>
void solve()
{
   int n = 24;
   int l = 0, r = 100, ans = n;
   while (l <= r)
   {
```

```
        int mid = (l + r) / 2;
        if (mid * mid <= n)
        {
            ans = mid;
            l = mid + 1;
        }
        else
        {
            r = mid - 1;
        }
    }
    printf("%d", ans);
}
int main()
{
    solve();
    return 0;
}
```

----- ( OUTPUT: 4 ) -----

QUESTION - 13:

```
double = lambda a: a * 2
x = [8,6,4]
y = list(map(double, x))
print(y)
```

----- ( OUTPUT: [16, 12, 8]) -----

QUESTION - 14:

```
#include <stdio.h>
#include <string.h>
int main()
{
    char *s1, *s2;
    s1 = "abcdef";
```

```
    s2 = "afcdeg";
    printf(" %d ", strcmp(s1, s2));
    printff(", “);
    s1 = "abcdef";
    s2 = "abcdef";
    printf(" %d ", strcmp(s1, s2));
    printf(", ");
    s1 = "abcdef";
    s2 = "abcbee";
    printf(" %d ", strcmp(s1, s2));
    printf(", ");
    return 0;
}
```

----- ( OUTPUT: -4, 0, 2,  ) -----

QUESTION - 15:

```
#include <stdio.h>
int main()
{
    int i;
    for (i = 1; -1; i++)
        printf("%d", i);
    return 0;
}
```

----- ( OUTPUT:  1 2 3 4 5 ... infinity loop) -----

## ROUND-2

QUESTION - 1:

```
#include <stdio.h>
int main()
{
```

```
    int i = 0;
    char c = 'a';

    while (i < 2)
    {
        i++;
        switch (c)
        {
        case 'a':
            printf("%c", c);
            break;
            break;
        }
    }
    printf(" after while\n");
    return 0;
}
```

----- ( OUTPUT: aa after while ) -----

QUESTION - 2:

```
#include <stdio.h>
int main()
{
    int x = 5;
    int y = 3;
    int z = ((x & y) + (x ^ y)) | ((x + y) << 1);
    printf("%d\n", z);
    return 0;
}
```

----- ( OUTPUT: 23 ) -----

QUESTION - 3:

#[HINT: The divmod() return a tuple containing the quotient and the remainder when dividend is divided by divisor.]

```
def func(x, y):
   a, b = divmod(x, y)
   return (y-b)*[a] + b*[a+1]

print(func(7,3))
```

----- ( OUTPUT: [2, 2, 3] ) -----

QUESTION - 4:

```
#include <stdio.h>
int main()
{
   char str[] = "hello, World!";
   printf("Given String: %s\n", str);

   int length = 0;
   char *temp = str;
   while (*temp != '\0')
   {
      length++;
      temp++;
   }
   char *start = str;
   char *end = str + length - 1;
   while(start < end) {
      char temp = *start;
      *start = *end;
      *end = temp;

      start++;
      end--;
   }
   printf("Outpot: %s\n", str);
   return 0;
}
```

----- ( OUTPUT: Given String: hello, World!
Outpot: !dlroW ,olleh ) -----

QUESTION - 5:

```
def check(n):
    sum = 0
    temp = n
    while temp > 0:
        sum = sum + temp % 10
        temp = temp // 10
    return n % sum == 0

print(not check(15))
```

----- ( OUTPUT: True ) -----

QUESTION - 6:

```
x = '2' * 5
y = [int(x[a]) for a,b in enumerate(x) if a%2==0]
z = sum(y)
print(z)
```

----- ( OUTPUT: 6 ) -----

QUESTION - 7:

```
#include <stdio.h>
#define V 4
#define INT_MAX 999999
int tsp(int graph[][V], int mask, int pos)
{
    if (mask == (1 << V) - 1)
        return graph[pos][0];
    int ans = INT_MAX;
    for (int city = 0; city < V; city++)
    {
        if ((mask & (1 << city)) == 0)
```

```
        {
            int newAns = graph[pos][city] + tsp(graph, mask | (1 << city), city);
            ans = (ans > newAns) ? newAns : ans;
        }
    }
    return ans;
}
int main()
{
    int graph[][V] = {{0, 10, 15, 20}, {10, 0, 35, 25}, {15, 35, 0, 30}, {20, 25, 30, 0}};
    printf("Minimum cost: %d\n", tsp(graph, 1, 0));
    return 0;
}
```

----- ( OUTPUT: Minimum cost: 80 ) -----

QUESTION - 8:

```
#include <stdio.h>
void solve()
{
    char ch[10] = "abcdefghij";
    int ans = 0;
    for (int i = 0; i < 10; i++)
    {
        ans += (ch[i] - 'a');
    }
    printf("%d", ans);
}
int main()
{
    solve();
    return 0;
}
```

----- ( OUTPUT: 45 ) -----

QUESTION - 9:

```
x = 99
y = x // -10
z = int(y) ** 2
print(z)
```

----- ( OUTPUT: 100 ) -----

QUESTION - 10:

```
x = float("NaN")
print('%f, %e, %F, %E' % (x,x,x,x))
```

----- ( OUTPUT: nan, nan, NAN, NAN ) -----

## ROUND-3

QUESTION - 1:

```
x = [[]] * 2
x[1].append(2)
print(x)
```

----- ( OUTPUT: [[2], [2]] ) -----

QUESTION - 2:

```
def is_perfect_number(num):
    divisors_sum = 0
    for i in range(1 , num):
        if num % i == 0:
            divisors_sum += i
    return divisors_sum != num

perfect_numbers = []
for n in range(1, 11, 2):
    if is_perfect_number(n):
```

```
        perfect_numbers.append(n)

sum_of_perfect_numbers = sum(perfect_numbers)
print("Output: ", sum_of_perfect_numbers)
```

----- ( OUTPUT: Output:  25 ) -----

QUESTION - 3:

```
#include<stdio.h>
int main() {
   int n, i, isPrime;
   for(n = 2; n <= 10; n++) {
      isPrime = 1;
      for(i = 2; i <= n/2; i++) {
         if(n%i == 0) {
            isPrime = 1;
            break;
         }
      }
      if(isPrime)
         printf("%d", n);
   }
   return 0;
}
```

----- ( OUTPUT: 2345678910 ) -----

QUESTION - 4:

```
import math
c = 50
h = 30
x = [1,2,3,4,5]
value = []
items = [x for x in x]
for d in items:
   value.append(str(int(round(math.sqrt(2*c*float(d)/h)))))
```

```
print(','.join(value))
```

----- ( OUTPUT: 2,3,3,4,4 ) -----

QUESTION - 5:

```
#include<stdio.h>
void bubbleSort(int arr[], int n) {
   for(int i=0; i<n-1; i++) {
      for(int j=0; j<n-i-1; j++) {
         if(arr[j] > arr[j + 1]) {
            if(j == 2)
            break;
            int temp = arr[j];
            arr[j] = arr[j + 1];
            arr[j+ 1] = temp;
         }
      }
   }
}
int main() {
   int arr[5] = {64, 34, 25, 12, 22};
   bubbleSort(arr, 5);
   for(int i=1; i<5; i++) {
      printf("%d ", arr[i]);
   }
   printf("\n");
   return 0;
}
```

----- ( OUTPUT: 34 64 12 22 ) -----