

Answer to the Question no-1

```
int n = 10;
int x = 5;
int index = 0;
int a[] = {12, 7, 3, 71, 2, 43, 38, 23, 45, 22};
int b[n];
for (int i=0; i<n; i++) {
    b[i]=a[i]+x;
}
```

Answer to the Question no-2

Given that

char s = “america”- The execution is wrong as “s” is a string & string has to be represented by the third bracket “[]”. So, the wrong thing is there is no third bracket after initializing the “s”. After that we have to use a semicolon to end the statement.

So, the correct answer is

```
char s[] = “america” ;
```

Answer to the Question no-3

After declaring the array[15](have to be long long int array[15]);, the next procedure for applying geometric progression is:

```
long long int array[15];
for(i=0;i<15;i++)
{
    array[i]=pow(2,i);
}
```

To apply this method we have to use #include<math.h> header file. Otherwise it will not work.

And another way is:

```
long long int array[15];
array[0]=1;
for(i=1;i<15;i++)
{
    array[i]=array[i-1]*2;
}
```

There is no need to use #include<math.h> header file.

Answer to the Question no-4

The question’s statement is

```

char a[10];
    a[0] = 98;
    a[1] = 97;
    a[2] = 'n';
    a[3] = 'a';
    a[4] = 'n';
    a[5] = 'a';
    a[6] = '\0';

```

Second statement is $a[0] = 98;$

Third statement is $a[1] = 97;$

As Fahim declared “a” is a character data type. So, it will take the character value of that integer.

We know that integer 98 means “b” in character and 97 means “a” in character.

As a result, when Fahim will print the value of string a in character format specifier “%c” $a[0]= 'b'$ and $a[1]= 'a'$ will happen.

After that it will print ‘banana’ . As $a[6]$ is a null character ,so it will stop after printing ‘banana’ .

Answer to the Question no-5

Given code is

```

char a[10];
char ch = 'a';
for(i = 0; i<8;i++)
{
    a[i] = ch+8-i;
}
a[8] = '\0';

```

When $i=0$, $a[0]=97+8-0=105 \rightarrow 'i'$

When $i=1$, $a[1]=97+8-1=104 \rightarrow 'h'$

When $i=2$, $a[2]=97+8-2=103 \rightarrow 'g'$

When $i=3$, $a[3]=97+8-3=102 \rightarrow 'f'$

When $i=4$, $a[4]=97+8-4=101 \rightarrow 'e'$

When $i=5$, $a[5]=97+8-5=100 \rightarrow 'd'$

When $i=6$, $a[6]=97+8-6=99 \rightarrow 'c'$

When $i=7$, $a[7]=97+8-7=98 \rightarrow 'b'$

If format specifier is ‘%d’ then $\rightarrow 105\ 104\ 103\ 102\ 101\ 100\ 99\ 98$

If format specifier is ‘%c’ then $\rightarrow i\ h\ g\ f\ e\ d\ c\ b$

So array ‘a’ will be stored with $\rightarrow i\ h\ g\ f\ e\ d\ c\ b$

Answer to the Question no-6

We know that the size of int is 4 bytes. If we declare an array with x elements then the size will be $4*x$ bytes.

In this case $x=12$

So the answer is $4*12=48$ bytes.

Example:

```
#include <stdio.h>
int main()
{
    int a[]={1,2,3,4,5,6,7,8,9,10,11,12};
    printf("%d",sizeof(a));
    return 0;
}
```

If we run this example we will get 48. It means that an int type array with 12 elements takes 48 bytes in memory.

Answer to the Question no-7

Lexicography comparison means to compare the alphabetic sequence letter by letter.

Example-If the first element of a sequence X is less than the first element of a sequence Y then X is lexicographically less than Y.

X=klodd, Y=foew.

If we compare this X1 is k, Y1 is f. In this case Y is lexicographically less than X.

For strings , we can use the bubble sort method to make lexicographic order.

Example:

“Cat”, “Rat”, “Chat”, “Hat”, “Fat”.

Primarily we have to compare index 0 of adjacent words.then index 1,2,3,... In this way we can replace it with actual order.

1- “Cat”, “Rat”, “Chat”, “Fat”, “Hat”

2- “Cat”, “Chat”, “Rat”, “Fat”, “Hat”

3- “Cat”, “Chat”, “Fat”, “Rat”, “Hat”

4- “Cat”, “Chat”, “Fat”, “Hat”, “Rat”.

Answer to the Question no-8

ASCII means American Standard Code for Information Interchange. ASCII value of a character means what kind of integer number is stored before a character value.

If we see examples, capital a(A) contains ASCII value 65 in integers. Small a contains ASCII value 97 in integers.

I can print the ASCII value of a character by changing the format specifier. If I use ‘%d’ as the format specifier, I can print the ASCII value of that character.

Example:

```
char v= 'a';
```

```
printf("%d",v);
```

It will print the ASCII value of a as 97.

Answer to the Question no-9

Bubble sorting means A method of arranging numbers in a series from small to large in which two numbers located next to each other can be rearranged in one or more steps if they do not correspond to the number line.

The given sequence is 12, 7, 9, 1, 3, 73, 11, 15, 62, 19, 4

If we run bubble sorting 5 times then the sequence will be

```
Sqne-1:7,12,1,9,3,73,11,15,19,62,4  
Sqne-2:7,1,12,3,9,11,73,15,19,4,62  
Sqne-3:1,7,3,12,9,11,15,73,4,19,62  
Sqne-4:1,3,7,9,12,11,15,4,73,19,62  
Sqne-5:1,3,7,9,11,12,4,15,19,73,62
```

Answer to the Question no-10

```
#include <stdio.h>  
#include <math.h>  
int main()  
{  
    long long int a;  
    scanf("%lld",&a);  
    int i,b,sum=0;  
    for(i=0;i>=0;i++)  
    {  
        if(a/(pow(10,i))>=1)  
        {  
            b=pow(10,i);  
            sum=sum+(a/b)%10;  
        }  
    }
```

```

    else
    {
        printf("%d",sum);
        break;
    }
}
return 0;
}

```

Answer to the Question no-11

```

#include <stdio.h>
int main()
{
    int n=3;
    int A[n+1][n+1];
    int i,j;
    for(i=1;i<4;i++)
    {
        for(j=1;j<4;j++)
        {
            scanf("%d",&A[i][j]);
        }
    }
    int a=0,b=0,c=0,d=0,e=0,f=0,g=0,x=0;
    for(i=1;i<4;i++)
    {
        if(A[1][i]>0)
        {
            a=a+A[1][i];
        }
        if(A[2][i]>0)
        {
            b=b+A[2][i];
        }
        if(A[3][i]>0)
        {
            c=c+A[3][i];
        }
    }
}
```

```

if(A[i][1]>0)
{
    d=d+A[i][1];
}
if(A[i][2]>0)
{
    e=e+A[i][2];
}
if(A[i][3]>0)
{
    f=f+A[i][3];
}
if(A[i][i]>0)
{
    g=g+A[i][i];
}
}
x=A[1][3]+A[2][2]+A[3][1];
if(a==b && b==c && c==d && d==e && e==f && f==g && g==x)
{
    printf("YES");
}
else
{
    printf("NO");
}

return 0;
}

```