Name :- TARUN

Reg.no.:- 19110483

Subject :- CSE101

Q-1

```
#include <stdio.h>
int main()
{
int n, j=0, i;
float sum=0, avg;
Printf ("enter the no. of element");
scanf ("%d", &n)
int a[n];
for (i=0; i<n; i++)
{
  scanf ("%d", &a[i]);
  if ((a[i]%2) != 0)
  { j++;
   sum = sum + a[i]
  }
}
avg = sum/j;
Printf ("%f", avg)
return 0.
}
```

```c
#include <stdio.h>

int main()

{

int n,j=0,i;

float sum=0,avg;
printf ("ënter the no of elements");

scanf("%d",&n); //input for no. of elements in array

int a[n];

for(i=0;i<n;i++)

{


scanf("%d",&a[i]); //get the array

if((a[i]%2)!=0)

{

j++; //total no. of odd no's in the array

sum=sum+a[i]; //sum of all odd no's in the array

}

}

avg=sum/j; //array average

printf("%f",avg);

printf("%d",&sum);

return 0;

}
```

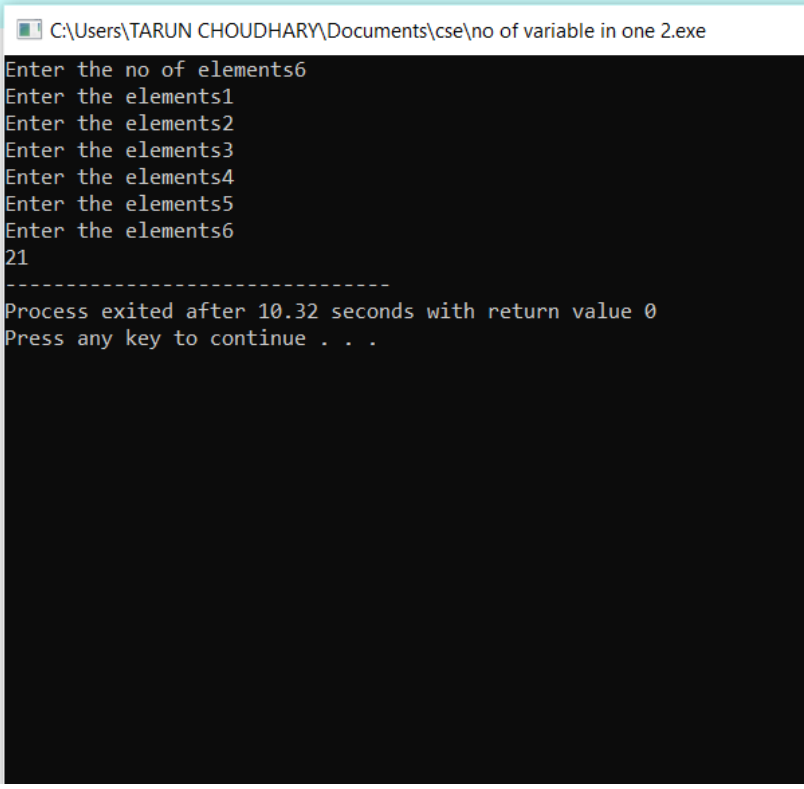
```
C:\Users\TARUN CHOUDHARY\Documents\cse\q1.exe
ðnter the no of elements5
1
3
2
5
3
3.0000006684108
--------------------------------
Process exited after 11.23 seconds with return value 0
Press any key to continue . . .
```

Q→2

```
#include <stdio.h>
int main()
{
int a[100], n, sum, o, i;
Printf("enter the no. of elements");
scanf( = ; &n; ++)
for (i=0 ; i<n ; i++)
{
Printf ("enter the elements");
scanf ("%d", & a[i]);
}
for (i=0, i<n; i++)
{
sum= sum + a [i] ;
Printf ("%d", sum);
return 0 ;
}
```

```c
#include<stdio.h>
int main()
{
int a[100],n,sum=0,i;
printf("Enter the no of elements");
scanf("%d",&n);

for(i=0;i<n;i++)
{
printf("Enter the elements");
scanf("%d",&a[i]);
}
for(i=0;i<n;i++)
{
sum=sum+a[i];
}
printf("%d",sum);
return 0;
}
```



Q→3

Sr no.	call by value	call by reference
1.	only o copy of the variable's value is passed the function	1. The variable value's address is passed to the function.
2.	The modifications made to the value of the passed variable present inside the function will be applicable to the fine only The original values of original values of actual parameters will remain uncharged even through the value of the formal parameter may be chayed.	2. change made to the value of variable inside the function will vallidate outside the function. as well. The original values of actual parameters will change when the value of the formal paromoter is attered.
3.	The formal and actual conjument will be created at different memory location	3. The formal and actual arguments will be created int the same memory location.

```
#include <stdio.h>
void swap (int *x, int *y)
{
    int temp;
    *x = *y;
    *y = temp;
}
int main()
{
    int a = 20;
    int b = 30;
    Printf (" before swapping value a and b are %d, %d \n", a, b);
    swap (&a, &b);
    Printf (" after swapping values a and b are: %d, %d \n", a, b);
    return 0;
```

RESULT :->

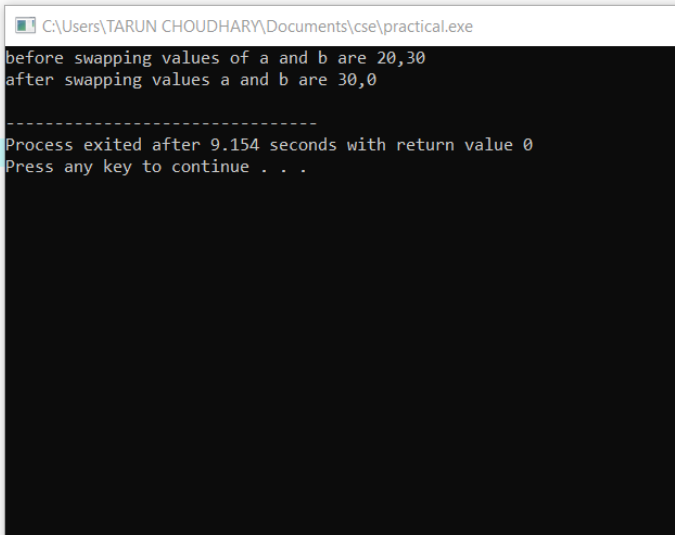
before swapping values of a and b are 20, 30

after swapping values a and b are 30, 20.

```c
#include <stdio.h>
void swap ( int *x, int *y)
{
    int temp;
    *x=*y;
    *y=temp;


}
int main()
{
    int a=20;
    int b=30;
    printf("before swapping values of a and b are %d,%d\n",a,b);
    swap ( &a , &b);
    printf ("after swapping values a and b are %d,%d\n ",a,b);
    return 0;

}
```



Q-4

Static	Extern.
⇒ The static storage class is used to declare an indetifier that is a function or a file and that exists and retains its value after control past from where it was declared. This storage class has a duration that is parmanent. A declared of the class retains its value from one call of function to the A variable is known only by the function to the next. the scope is local. A variable is known only by the function it is globally in the file it is on only by the function with in that file. this storage class gaurantees that declaration of the variable also initially the variable to zero	The extern storage class is used to declare a global variable that will be known to the func in a file and capable of being known to all functions in A Programm. This permanent Any variable of this class retains of this class retains its value until changed by another. alignement the scope is global. A variable can be known in seen by all function with in a Program.

Register values tell the compiler to store the variable in CPU register instead of memory frequently used variable are kept in register and they have faster accessibility we can never get the address of there variable register keyword is used to to declare the register values.

Scope :- The are local to the function.

Default value :- Default initialized value is the garbage value

life time :- Till the end of the execution of the locle in which it is defined.

Example :-

```
#include <stdio.h>
int main()
{
    register char n = 'S';
    register int a = 10;
    auto int b = 8;
    Printf("The value of register variable b: %c\n",n);
    Printf("The sum of auto and register variable: %d\n",(a+b));
    return 0;
}
```