Department of Computer Science and Engineering
Amrita School of Engineering, Bengaluru
Topic: Functions-Paasing Array, Recursive,Nested,StorageClass

Practice Questions:

1.Programs discussed in the class:(refer function notes, passing array to a function)

1. Display Array using function , function passing as an argument to an array

```
#include<stdio.h>

int display(int marks[]);

int main()
{
  // float avg;
    int marks[5] = {99, 90, 96, 93, 95};
    display(marks);       // name of the array is passed as argument.   base address: array name it self is a base address or marks[0]

}
int display(int marks[])
{
    int i;
// sum = 0;
    //float avg;
    for (i = 0; i <= 4; i++) {
printf("%d\t",marks[i]);


 //return marks; // not allowed
}
}
//call by value---actual value
```

2. Linear Search program with functions

```
#include<stdio.h>
  int linear(int a[] ,int n ,int key);
int main()
{
    int a[10],i,key,n;
    printf("How many elements?");
    scanf("%d",&n);


    printf("Enter array elements:n\n");
    for(i=0;i<n;++i)
```

```
        scanf("%d",&a[i]);

    printf("Enter element to search:\n");
    scanf("%d",&key);

linear(a,n,key);


}
int linear(int a[],int n,int key)
{
int i;

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

        if(a[i]==key)
//count++;

            break;

    if(i<n)
        printf("Element found at index %d",i);
    else
        printf("Element not found");

    return 0;
}
```

3. Function to find the square

```
 #include<stdio.h>
int square(int);
int main()
{
        int num, res;
       printf("Enter a number\n");
       scanf("%d", &num);
       res=square(num);
       printf("Square of %d  = %d", num, res);
}

int  square(int x)
{
         return  (x*x);
}
```

4. Decimal to Binary with functions

```
#include <stdio.h>
#include <math.h>

long decimalToBinary(int decimalnum)
{
   long binarynum = 0;
   int rem, temp = 1;

   while (decimalnum!=0)
   {
      rem = decimalnum%2;
      decimalnum = decimalnum / 2;
      binarynum = binarynum + rem*temp;   //  1 1 1 0 1
      temp = temp * 10;
   }                                          //decimal number  12       12/2, 6, rem=0       ,      rem= 0,
3 , rem=1, 3/2=1,  //rem=1,                1100
   return binarynum;
}

int main()
{
   int decimalnum;
   printf("Enter a Decimal Number: ");
   scanf("%d", &decimalnum);
//bin=decimalToBinary(decimalnum);
//printf("%ld",bin);
   printf("Equivalent Binary Number is: %ld", decimalToBinary(decimalnum));
   return 0;
}
```

Extra questions

5. Finding average of marks by passing array to a function.
6. Write a C program to find out maximum element in the given array. The function should accept an array and return the maximum value to the calling function.
7. Write C program to add two matrices by array and rows, columns to the function as an argument
8. Write a C program to find out row sum and column sum by passing array to function.
9. Write a C program for bubble sort, selection sort , insertion sort by passing array.

Nested Functions

1. Finding average and printing array

```
 #include<stdio.h>
float findAverage(int marks[]);
int printarray(int marks[]);
int main()
{
```

```
    float avg;
    int marks[5] = {99, 90, 96, 93, 95};
printarray(marks);
printf("program done\n");

  // avg=  findAverage(marks);       // name of the array is passed as argument.  marks=marks[0];
base address: array name it self is a base //address or marks[0]
 // printf("Average marks = %.1f", avg);
  //  return 0;
}

int printarray(int marks[])
{
int i;
float avg;
for (i = 0; i <= 4; i++)
printf("%d\t",marks[i]);
avg=findAverage(marks);
printf("Average marks = %.1f", avg);
printf("average done\n");

}

float findAverage(int marks[])
{
    int i;
float sum = 0;
    float avg;
    for (i = 0; i <= 4; i++) {
        sum += marks[i];
    }
printf("%f\n",sum);
    avg = (sum / 5);
 //printf("Average marks = %.1f", avg);
//printf("%f",avg);
   return avg;
}
```

2. Write a C program with nested functions, where one function sum_array() computes the sum of elements of the 1D array and sum_array() call printsum() and print the sum and the printsum() can call count() function and count the number of elements present in the sum and return to the printsum().

Recursion:

1. Write a program to find factorial using recursion

```
#include<stdio.h>
```

```
int find_factorial(int n);
int main()
{
  int num, fact;
  //Ask user for the input and store it in num
  printf("\nEnter any integer number:");
  scanf("%d",&num);    //5

  //Calling our user defined function

  fact =find_factorial(num);     //function call

  //Displaying factorial of input number
  printf("\nfactorial of %d is: %d",num, fact);
 // return 0;
}
int find_factorial(int n)
{
  //Factorial of 0 is 1
  if (n==0)

    return(1);  //return constant

  //Function calling itself: recursion
  return(n*find_factorial(n-1));
}

//return (5*find_factorial(4))    or 5*4!        return(5*24)= 120                  n*(n-1)
5!=n*n-1, 5*4*3*2*1=120
//return (4*find_factorial(3))   or 4*3!   retuen(4*6)=24

//return (3*find_factorial(2))  or 3*2!       //return 3*2  =6
//return (2*find_factorial(1)) or 2*1!        ///return (2*1) =2
//return (1*find_factorial(0)) or 1*0!        //   return 1*1=1
//return 1
```

2. Write a program in C to Print Fibonacci Series using recursion.

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

```
{
        int n, i;
        printf("Enter the number of element you want in series :\n");
        scanf("%d",&n);
        printf("fibonacci series is : \n");
        for(i=0;i<n;i++)
        {
                printf("%d ",fibonacci(i));
        }
}

int fibonacci(int i)
{
        if(i==0)
           return 0;
        else if(i==1)
           return 1;
        else
           return (fibonacci(i-1)+fibonacci(i-2));
}
```

3. Write a program in C to calculate the sum of numbers from 1 to n using recursion.
4. Write a program in C to find the sum of digits of a number using recursion.
5. Write a program in C to print the array elements using recursion.

Storage class programs

1. Auto:

```
#include <stdio.h>
int main()
{
  auto int j = 1;
//auto int j ;
  {

    auto int j= 2;
    {

 auto int j = 3;

      printf ( " %d ", j);    //j=3     j=1, j=2, j=3
}

    printf ( "\t %d ",j);
  }

  printf( "%d\n", j);}
```

Observe the output by changing the brackets.

2. Static:

```
#include <stdio.h> /* function declaration */

void next();
void display();
static  int counter = 7; /* global variable */


int main()
 {
 while(counter<10) {

      next();
      counter++;   }

display();
//return 0;
}

void next() {    /* function definition */

   static int iteration = 13; /* local static variable */
```

```
  iteration ++; //14

  printf("iteration=%d and counter= %d\n", iteration, counter);
}


void display()
{
printf("%d",counter);
}
```

Remove static and observe the output , analyse both the outputs.

3. Extern

storage_variable.h

```
extern int num1=5 ;
extern int num2 =6;
```

storage_exter.c

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

printf("%d\n",num1);
printf("%d\n",num2);
}
```

Store it a stwo different files and understand the usage of extern in two different files.

Practice question:

static:

```
#include <stdio.h>
 /* Function declaration */ void display();
 int main()
 { display();
 display();
 return 0; }
/* Function definition */
 void display()
 { int n1  = 10; /* static variables are declared only once */
 static int n2  = 10;
 printf("Local n1 = %d, Static n2 = %d\n", n1, n2);
n1++; // Increment local variable
 n2++; // Increment static variable
 }
```

extern

Let us write two programs to demonstrate how to share variables and functions among various C programs using `extern` keyword.

First write and save as **stoarge_extern_file1.c**

```
#include <stdio.h>

// Global variable
int num;

void display()
{
   int i;
   for(i=1; i<=num; i++)
   {
      printf("num = %d\n", i);
   }
}
```

Now , write main program and save it as storage_main.c

```
/* Declare link to variable defined in stoarge_extern_file1.c */
extern int num;

/* Declare link to function defined in stoarge_extern_file1.c  */
extern void display();

int main()
{
   // Access external variable
   num = 5;

   // Access external function
   display();

   return 0;
}
```

Compile using the following command

```
gcc stoarge_extern_file1.c storage_main.c main
./main
```

Example: Program to illustrate the properties of a static variable.

```
#include<stdio.h>
void incre();
int main()
 { int i;
 for (i=0; i<3; i++)
 incre();
}
void incre()
 { int avar=1;
 static int svar=1;
 avar++;
 svar++;
 printf("\n Automatic variable value : %d",avar);
 printf("\t Static variable value : %d",svar);
      }
```