Department of Computer Science and Engineering
Institute of Technical Education & Research, SOA, Deemed to be University

MINOR ASSIGNMENT-05

Practical Programming with C (CSE 3544)

Publish on: 06-12-2024 **Submission on:** 08-12-2024
Course Outcome: CO_3 **Program Outcome:** PO_3 **Learning Level:** L_4

Problem Statement:

Experiment with pointers and dynamic memory allocation in C.

Assignment Objectives:

To learn how to manipulate arrays using pointers and to learn `malloc`, `mcalloc`, `realloc` & `free` to allocate and free dynamic memory.

Answer the followings:

1. Consider the following ANSI C program;

```
#include<stdio.h>
int main(){
  int arr[4][5],i,j;
  for(i=0;i<4;i++){
    for(j=0;j<5;j++){
     arr[i][j]=10*i+j;
   }}
 printf("%d\n",arr[2][4]);
 printf("%d\n",*(*(arr+2)+4));
 return 0;}
```

What is the output of the above program?

Output with explanation

24
24
*(arr+2) points to address of arr[2]
((arr+2)+4)) points to arr[2][4] i.e 24

2. Consider the following ANSI C program;

[GATE 2021]

```
#include<stdio.h>
int main(){
  int arr[4][5],i,j;
  for(i=0;i<4;i++){
    for(j=0;j<5;j++){
     arr[i][j]=10*i+j;
   }}
 printf("%d\n",*(arr[1]+9));
 return 0;}
```

What is the output of the above program?

Show the 2-D array and the output

24
Arr = {0,1,2,3,4}
{10,11,12,13,14}
{20,21,22,23,24}
{30,31,32,33,34}

3. Consider the following C program

[GATE 2020]

```
#include<stdio.h>
int main(){
int a[4][5]={
       {1,2,3,4,5},
       {6,7,8,9,10},
       {11,12,13,14,15},
       {16,17,18,19,20}};
 printf("%d\n",*(*(a+*a+2)+3));
 return 0;}
```

The output of the program is

Output▼

14

4. Write the output of the following program? Assume that the base address of a given array **a** is 1000?

```
int main(){
int a[3][3]=(4,5,6,7,8,9,1,2,3);
printf("%p %p %p\n",a[1]+2,*(a+1)+2,&a[1][2])
   ;
printf("%d %d %d\n",*(a[1]+2),*(*(a+1)+2), a
   [1][2]);
return 0;
}
```

Output▼

0061FF10 0061FF10 0061FF10

9 9 9

5. Select the output of the following program.

```
int main(){
 int a[][3]=(4,5,6,7,8,9,1,2,3);
 printf("%d,", *a[2]);
 printf("%d,", a[2][0]);
 printf("%d ", **(a+1+('b'-'a')));
 return 0;
}
```

ASCII value of a=97 and b=98

Output▼

(A) 1024,1,1 (C) 1024,2,1024

(B) 1,1,1 (D) None of these

6. Find the output of the code snippet

```
int main(){
  int a[][2][4]=(5,6,7,8,9,11,12,1);
  printf("%d\n",*(*(*(a+0)+1)+2));
  return 0;
}
```

Output▼

12

7. Describe the output for the following code snippet.

```
void fun(int arr[][3]){
  printf("%d\n",*(*(arr+2)+1));
  printf("%p\n",(*arr)+2);
  printf("%p\n",&arr[0][2]);
  printf("%d\n",*(((*arr)+1)+1));
}
int main(){
  int a[][3]=(5,6,7,8,9,4,3,2,1);
  fun(a);
  return 0;
}
```

Output▼

2
0061FF04
0061FF04
7

8. Explain the below declaration(s).

```
(1) int process(int (*pf)(int a, int b)) ;
(2) int (*fun(int, void (*ptr)()))();
(3) int *(*p)(int (*a)[ ]);
(4) int (*p)[10];
(5) float *p[20];
(6) int p(char *a);
(7) int (*p(char * a))[10];
(8) int * (*p [10]) (char  *a);
```

Output▼

1) pf is a pointer to function, int a, int b are arguments.

2) fun is a function (it also acts a pointer to it), int and ptr are arguments.

3) p is a pointer to a function that takes a pointer to an array of integers

4) p is a pointer to an array of 10 integers

5) p is a pointer to 20 floats.

6) p is a function that takes a pointer to char and returns int.

7) p is a function that takes pointer to char and returns a pointer to an array of 10 integers.

8) p is an array of 10 pointers to functions.

9. What is printed by the following ANSI C program? **[GATE 2022]**

```
#include<stdio.h>
int main(void){
 int x = 1, z[2] = {10, 11};
 int *p = NULL;
 p = &x;
 *p = 10;
 p = &z[1];
 *(&z[0] + 1) += 3;
 printf("%d, %d, %d\n", x, z[0], z[1]);
 return 0;}
```

Output with explanation

(A) 1, 10, 11 (C) 10, 14, 11

(B) 1, 10, 14 (D) 10, 10, 14

*p points to z[1] (11)

10. Find the output and different types of pointer involved in the code snippet;

```
int main(){int *p=NULL;
  p=(int *)malloc(sizeof(int));
  *p=10;
  free(p);
  int *q;
  q=(int *)malloc(sizeof(int));
  *q=15;
  printf("%d  %d\n",*p,*q);
  return 0;}
```

Output▼

15 15

11. State the output of the following program. Assume the address of p is 1000 and q is 2000.

```
#include<stdio.h>
#include<stdlib.h>
void fun(int **q);
int main(){
 int *p=(int *)malloc(sizeof(int));
 *p=55;
 fun(&p);
 printf("%d %p\n",*p,p);
 return 0;
}
```

```
void fun(int **q){
  int r=20;
  **q=r;
  printf("%p\n",*q);
}
```

Output▼

00BA1598
20 00BA1598

12. Select the desire output of the following code snippet with reason:

```
int *fun();
int main(void){
  int *ptr;
  ptr=fun();
  printf("%d\n",*ptr);
  return 0;
}
```

```
int *fun(){
 int a=10,b=20;
 int sum=0;
 sum=sum+a+b;
 return &sum;
 }
```

Output with reason▼

(X) Unexpected behavoir
(C) 30
(B) Address of sum
(D) None of these

*ptr results in (A) as memory is not referenced

13. Select the desire output of the following code snippet with reason:

```
int *fun();
int main(void)
{
  int *ptr=fun();
  printf("%d\n",*ptr);
  return 0;
}
```

```
int *fun(){
 int a=10,b=20,*sum;
 sum=(int *)malloc(
    sizeof(int));
 *sum=a+b;
 return sum;
 }
```

Output with reason▼

(A) Unexpected behavoir
(✓) 30
(B) Address of sum
(D) None of these

*sum = 30

14. Find the output of the following program.

```
int main(){int *ptr;
 ptr=(int *)realloc(NULL,sizeof(int));
 *ptr=100;
 printf("%d\n",*ptr);
 return 0;}
```

Output▼

100

15. Write the output of the following program.

```
1  int main(){int *ptr;
2   ptr=(int *)calloc(1,sizeof(int));
3   *ptr=100;
4   printf("%d\n",*ptr);
5   ptr=(int *)realloc(ptr,0);
6   ptr=NULL;
7   printf("%p\n",ptr);
8   return 0;}
```

Output▼

Output at line-4: 100

Output at line-7: 00000000

Line number-6 can be treated as like `free()` to deallocate memory-Y|N

16. Consider the following code segment:

```
int main(){int b=65;
void p=b;
 printf("%d",p);return 0;}
```

Observation▼

variable can't be declared void

17. Select the output of the following program.

```
int main(){
 int b=65;
 void *p=&b;
 int *j=(int *)p;
 char *ch=(char *)p;
 printf("%d  %c\n",*j,*ch);
 return 0;
}
```

Output▼

(A) 65 65 (C) Compile time error

(B) 65 A (D) Run time error

18. Write the output of the code snippet. Also show the stack and heap memory for this application.

```
int main(){int i;
 int *p=(int *)malloc(sizeof(int));
 *p=100;
 p=(int *)malloc(5*sizeof(int));
 for(i=0;i<5;i++){
    scanf("%d",p+i); /* 10,20,30,40,50 */
 }
 for(i=0;i<5;i++){
  printf("%d...%d\n",p[i],*(p+i));
 }
 return 0;}
```

Output▼

10 ... 10
20 ... 20
30 ... 30
40 ... 40
50 ... 50

19. Write the output of the code snippet. Also show the stack and heap memory for this application.

```
int main(){
int i,*p,*rp;
p=(int *)malloc(5*sizeof(int));
for(i=0;i<5;i++)
   scanf("%d",p+i); /* 10,20,30,40,50 */
rp=(int *)realloc(p,10*sizeof(int));
for(i=5;i<10;i++)
   scanf("%d",rp+i);/* 9,8,6,5,4 */
for(i=0;i<10;i++){
 printf("%d...%d\n",rp[i],*(rp+i));
}
return 0;}
```

Output▼

10 ... 10
20 ... 20
30 ... 30
40 ... 40
50 ... 50
9 ... 9
8 ... 8
6 ... 6
5 ... 5
4 ... 4

20. Which of the following statements are true?.

```
(1) (void *)0 is a void pointer
(2) (void *)0 is a NULL pointer
(3) int *p=(int *)0; p is a NULL pointer
(4) a[i]==i[a]
(5) a[i][j]== *(*(a+i)+j)
```

Output▼

True True
True
True
True

21. State the output of the code.

```
#include<stdio.h>
int f(int n){
 while(--n>=0){
  printf("%d ",n-2);}
 return 1;}
```

```
int main(){
  int (*p)(int)=f;
  (*p)(8);
  return 0;}
```

Output▼

5 4 3 2 1 0 -1 -2

22. Which of the given statements about the following code snippet is/are correct?

```
void fun(){
   int *q=(int *)malloc(sizeof(int));
   *q=20;
}
int main(){
 int *p;
 int *r=NULL;
 fun();
 return 0;
}
```

```
(i)   p is a wild pointer
(ii)  r is a NULL pointer
(iii) q is dangling pointer
(iv)  p is dangling pointer
(v)   fun() is making memory leak
```

Output▼

i) ii) iii) v) True

iv) False

23. Check the error or output of the following program?

```
int main(){
void *p;
int *i=20;
p=&i;
void *q=p; //line-4
//line-5
printf("%d %d %d\n",i,*p,*q);
}
```

(i) 20 20 20
(ii) 20 30 20
(iii) compile error at line-4
(iv) compile error at line-5

Output▼

Compile error core at line-5

24. Write the output of the given code snippet.

```
#include<stdio.h>
int main(){
   void demo();
   void (*fun)();
   fun=demo;
   (*)fun();
   fun();
   return 0;
}
```

```
#include<stdio.h>
void demo(){
 printf("SS");
}
```

Output▼

SSSS

25. Write the output of the given code snippet that uses pointer to function or function pointer.

```
int fun(int x,int y){
  int z=x+y+x*y;
  return z;
}

#include<stdio.h>
int main(){
  int (*fun_ptr)(int,int);
  fun_ptr=fun;
  int x=fun_ptr(34,56);
  printf("%d\n",x);
  return 0;
}
```

Output▼

1994

26. Mention the output of the following code snippet. [Array of pointers to function returning int type].

```
#include<stdio.h>
int main(){
int x,y;
int (*fun_ptr[2])(int,int);
fun_ptr[0]=fun1;
x=fun_ptr[0](4,5);;
fun_ptr[1]=fun2;
y=(*fun_ptr[1])(4,5);
printf("%d...%d\n",x,y);
return 0;
}
```

```
int fun1(int x,int y){
 return x+y;
}

int fun2(int x,int y){
 return x*y;
}
```

Output▼

9....20

27. Find out the correct syntal(s) for making a constant pointer (i.e. The value of the pointer is constant and pointer cannot be modified).

```
(1) const <data_type> * ptr;
(2) <data_type> * const ptr;
(3) <dat_type> const *ptr;
(4) <data_type> const * const fun_ptr
(5) None of these
```

Output▼

(2) <data_type> * const ptr;

28. Find out the correct syntal(s) for a pointer to constant (i.e. The pointer cannot able to change the value of the variable/array that it points).

```
(1) const <data_type> * ptr;
(2) <data_type> * const ptr;
(3) <dat_type> const *ptr;
(4) <data_type> const * const fun_ptr
(5) None of these
```

Output▼

(1) Const <data_type> *ptr;
(3) <data_type> const *ptr;

29. Select the correct way of declaring and initializing pointer to function (i.e. function pointer).

```
(1) int (*ptr)(int,int,int)=funname;
(2) int *ptr(int,int,int)=funname;
(3) int (*ptr)(int,int,int)=&funname;
(4) (int *) ptr(int,int,int)=funname;
(5) None of these
```

Output▼

int (*ptr) (int,int,int) = funname;
int (*ptr) (int,int,int) = &funname;

Department of Computer Science and Engineering
Institute of Technical Education & Research, SOA, Deemed to be University

MINOR ASSIGNMENT-06

Practical Programming with C (CSE 3544)

Publish on: 07-12-2024 **Submission on:** 09-12-2024
Course Outcome: CO_4 **Program Outcome:** PO_5 **Learning Level:** L_5

Problem Statement:

Working with different storage classes and Experiment with one of the powerful tool, recursion, in problem solving and programming.

Assignment Objectives:

To learn about storage classes and get the idea of how function calls itself to solve computational problem.

Answer the followings:

1. Consider the following ANSI C program;

```
#include <stdio.h>
int main()
{
        static int i=5;
        if(--i){
                main();
                printf("%d ",i);
        }
        return 0;
}
```

What is the output of the above program?

Output with explanation

0 0 0 0

recursive call to main() until i=<0. So, output is 0 0 0 0

2. Consider the following ANSI C program;

```
#include <stdio.h>
int a, b, c = 0;
void prtFun(void);
int main()
{
       static int a = 1; /* Line 1 */
        prtFun( );
        a+=1;
        prtFun( );
        printf("\n %d %d ", a, b);
        return(0);
}
void prtFun(void)
{
        static int a = 2; /* Line 2 */
        int b = 1;
        a + = ++b;
        printf(" \n %d %d ", a, b);
}
```

What is the output of the above program?

Output with explanation

4 2
6 2
2 0

First call to prtFun()
b = 1
b = 2, a += 2, a = 4
printf(4 2)

Second call
printf(6 2)

2 0 (as b not assigned)

3. Consider the following ANSI C program:

```
#include <stdio.h>
int a, b, c = 0;
void prtFun(void);
int main()
{ auto int a = 1; /* Line 1 */
        prtFun( );
        a+=1;
        prtFun( );
        printf("\n %d %d ", a, b);
        return(0);
}
void prtFun(void)
{ register int a = 2; /* Line 2 */
        int b = 1;
        a + = ++b;
        printf(" \n %d %d ", a, b);
}
```

What is the output of the above program?

Output with explanation

This program will result in a compilation error because a register variable cannot have its address taken.

4. What is printed by the following ANSI C program? **[GATE 2005]**

```
#include<stdio.h>
int f(int n, int k){
        if(n==0) return 0;
        else if(n%2) return f(n/2, 2*k)+k;
        else return f(n/2, 2*k)-k;
}
int main(){
        printf( "%d",f(20,1));
        return 0;
}
```

Output with explanation

16
8
12
10
9

Output: 9

For each call, n is halved and k is multiplied by 2

5. What is printed by the following ANSI C program? **[GATE 2007]**

```
#include<stdio.h>
void f(int n){
        if(n<=1){
                printf("%d",n);
        }
        else{
                f(n/2);
                printf("%d",n%2);
        }
}
int main()
{
        f(173);
        return 0;
}
```

Output with explanation

173 into binary

10101101

f recursively divides n by 2,

[GATE 2004]

6. Consider the following C function:

```
int f(int n)
{
   static int i = 1 ;
   if (n >=5) return n;
   n = n+1;
   i++;
   return f(n);
}
```

The value returned by `f(1)` is
(A) 5 (B) 6 (C) 7 (D) 8

Show the execution pattern▼

f(1) → f(2) → f(3) → f(4)
↓
f(5)

output
5

GATE-2019

7. Consider the following C program

```
#include<stdio.h>
int r(){
  static int num=7;
  return num--;
}
int main()
{
        for(r();r();r())
           printf("%d ", r());
       return 0;
}
```

Which one of the following values will be displayed on execution of the program? (A) 41 (B) 52 (C) 63 (D) 630

Trace the execution and it's output▼

output
1 3 9 1
r() → 7
r() → 6
r() → 5

8. The integer value printed by the ANSI-C program given below is :

```
int funcp(){
  static int x = 1;
  x++;
 return x;
}
int main(){
  int x,y;
  x = funcp(); x=2
  y = funcp()+x; y = 3+2 = 5
  printf("%d\n", (x+y)); 2+5=7
  return 0;
}
```

Output▼

7

9. Consider the following C program

Find the error in the program with proper reasoning

```
#include<stdio.h>
int main(){
        register int a =10;
        int *ptr = NULL;
        ptr = &a;
        *ptr = 5;
        printf("%d",*ptr);
        return(0);
}
```

Output▼

Cannot take address of register variable.

10. Consider the following C function;

file1.c

```
extern in count;
void write_extern(){
        count +=2;
}
```

file2.c

```
#include<stdio.h>
#include "file1.c"
int count = 5;
int main(){
        write_extern();
        write_extern();
        printf("%d\n", count);
        return(0);
}
```

Find the output if "`file2.c`" is compiled and executed:

Output with explanation▼

9

* Each call to write_extern() increments count by 2.

11. Write the output of the following program;

```
#include<stdio.h>
int i=5;
int main()
{
        extern int j;
        printf("\ni=%d \nj=%d",i,j);
        int j=10;
        return 0;
}

int j =10;
```

Output▼

Error : Redeclaration of J as Global variable.

12. Write a program to find the sum of an array elements using recursion.

Program and Output▼

```
#include <stdio.h>
int sum (int arr[], int n){
        if (n<=0) return 0;
        return arr[n-1]+ sum(arr, n-1);
}

int main(){
int arr[]=
    {1,2,3,4,5};
printf("sum=%d",
    sum(arr,5));
return 0;
} //Sum = 15
```

13. Write a program to print "n" Fibonacci numbers using recursion.[N.B: The program format should be as follows]

```
#include <stdio.h>
... print_fibo(.....){
        ...
        ...
}
... main(){
        // get data from user
        print_fibo(...); // to print elements
}
```

Program and Output▼

```
#include <stdio.h>
void print_fibo (int n) {
        static int a=0, b=1, next;
          if (n>0) {
              printf("%d ", a);
              next = a+b;
               a=b;
               b = next;
               print_fibbo(n-1);
          }
    }
int main() {
        int n;
        printf("Enter numbers:");
        scanf("%d", &n);
        print_fibo(n);
         return 0;
}
```

Output

```
Enter numbers: 5
0 1 1 2 3
```