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

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

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▼

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▼

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▼

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▼

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▼

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

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▼

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▼

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▼

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

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 (C) 30
(B) Address of sum (D) None of these

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▼

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:

Output at line-7:

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▼

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
(B) 65 A
(C) Compile time error
(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▼

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▼

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▼

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▼

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

```
(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
```

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

Output▼

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▼

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▼

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▼

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

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

Output▼

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).

Output▼

```
(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
```

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).

Output▼

```
(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
```

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

Output▼

```
(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
```