1)

```
#include<stdlib.h>
struct Test
{
int a;
struct Test *p;
};
int main()
{
struct Test *pt1=malloc(sizeof(struct Test));
pt1->a=10;
pt1->p=pt1;
printf("%d %d %d %d\n",pt1->a, pt1->p->a,pt1->p->p->a,pt1->p->p->p->a);
free(pt1);
}
```

- Error in line: free(pt1)
- Error in line: struct Test *pt1=malloc(sizeof(struct Test));
- Error in line: pt1->p=pt1;
- No Error. Displays: 10 10 10 10

2) 1. **Consider the following 3 C functions**

```
//P1
int * g (void)
{
int x = 10;
return (&x);
}
//P2
int * g (void)
{
int * px;
*px = 10;
return px;
}
//P3
int *g (void)
{
int *px;
px = (int *) malloc (sizeof(int));
*px = 10;
return px;
}
```

Which of the above three functions are likely to cause problems with pointers?

- Only P3
- Only P1 and P3
- Only P1 and P2
- P1, P2 and P3

3) 1. **Predict the output of the below code**

```
#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("%dn", *(*(a+**a+2)+3));
return(0);
}
```

- 14
- 20
- 18
- 19

4) **Return type of free function is**

- void
- void*
- int
- starting address of the memory

5) 1. **What does the following function print for n = 24?**

```
void fun(int n)
{
if (n == 0)
return;
printf("%d", n%2);
fun(n/2
}
```

- 00000
- 00011
- 11000
- 11111

Back to Units

Class2_Overview of static Memory Allocatio