CS270 – Dynamic memory Name_____________________________________

1) What does the following code print:

```
#include <stdio.h>
int main() {
    int arr[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
    int *ptr = &arr[0];
    while (ptr <= &arr[3]) {
        printf("%d ", *ptr);
        ptr++;
    }
    printf("\n");
}
```

2) Consider the following program, where are i, j and k are stored in memory?

```
int i;
int main()
{
    int j;
    int *k = (int *) malloc (sizeof(int));
}
```

3) What is the output?

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

void fun(int *a)
{
    a = (int*)malloc(sizeof(int));
}

int main()
{
    int *p;
    fun(p);
    *p = 6;
    printf("%dn",*p);
    return(0);
}
```

4) What is wrong with the following code?

```
#include<stdio.h>
int main()
{
    int *p = (int *)malloc(sizeof(int));
    p = NULL;
    free(p);
}
```

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

```
[PI] 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;
}
```

6) What does the following code print?

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

int main()
{
    int r = 3, c = 4;
    int *arr = (int *)malloc(r * c * sizeof(int));

    int i, j, count = 0;
    for (i = 0; i <  r; i++)
      for (j = 0; j < c; j++)
         *(arr + i*c + j) = ++count;

    for (i = 0; i <  r; i++)
      for (j = 0; j < c; j++)
         printf("%d ", *(arr + i*c + j));

   /* Code for further processing and free the
      dynamically allocated memory */

   return 0;
}
```