Lab 5

You are expected to copy and paste your code into each corresponding box in this handout and submit it as a Word document or PDF file before the due. Additionally, your lab instructor will tell you which three questions you must showcase during the lab session. While you may demonstrate your code running in person after the due date, your file must be submitted on time.

Hint #1: For questions asking for 'a **function** which takes in xyz', the xyz refer to function parameters, not the use of scanf().

Hint #2: For questions asking for 'a **program** which reads in xyz', the xyz refer to the use of scanf().

Hint #3: For questions asking for 'a program', you are expected to include all necessary #include and define your main function. It's up to you whether to define additional helper functions, though usually, you don't need to.

Task 1: To define a **program** to print your machine pointer size

```
#include <stdio.h>

int main(void) {
     int *p = 0;
     printf("%lu\n", sizeof(p));
     return 0;
}
```

Task 2: To define a **program** to read an integer from user, using a pointer

[Hint 1: besides of the integer (n), you need to define a pointer to integer (p)]
[Hint 2: in your scanf(), only p got shown, NOT n]

```
#include <stdio.h>

int main(void) {
     int n;
     int* p = &n;
     printf("Enter an integer:\n");
     scanf("%i", p);

     printf("You entered: %i\n", *p);

     return 0;
}
```

Task 3: define a **program** with an int variable (n) and a pointer (p) points to n, and then print the value and the address of n, using p

[Hint 1: in your printf(), only p got shown, NOT n]
[Hint 2: you may decide what's the value in n]

```
#include <stdio.h>

int main(void) {
     int n = 0;
     int* p = &n;

     // compiler wants to use %p instead of %d
     printf("%p\n", p);
     return 0;
}
```

Task 4: To define a **program** which holds an integer array, called data, with initial values as {11, 12, 13}, with a pointer points to data[0]; use the pointer to updates data[1] to 60, and print data[1] using the pointer

[Hint: it's your choice whether move the pointer or not, either way works]

```
#include <stdio.h>

int main(void) {
     int data[3] = {11, 12, 13};
     int* dptr = data;
     *(dptr + 1) = 60;

     printf("%i\n", *(dptr + 1));

     return 0;
}
```

Task 5: To define a **program** which reads in a string from user, and then print it with the third char got replaced by '?'

Example: (no need to put in or print the double quos)

input:	"hello"
output:	"he?lo"

[Hint: you should use a char[], not a pointer. Consider why is it]

```
#include <stdio.h>

int main(void) {
     char input[32];
     printf("Enter a string (max 32 characters):\n");
     scanf("%s", input);
     input[2] = '?';
     printf("%s\n", input);

     return 0;
}
```

Task 6: define a **function** to swap 2 integers and to return the bigger value, with inputs as two pointers to integer

```
int swap (int* x, int* y);
```

Usage :

```
int main(void) {
        int m = 22, n = 11;
        printf("%d", swap (&m, &n));   /*should print 22*/
        printf("%d", m);               /*should print 11*/
}
```

```
int swap(int* x, int* y) {
     int hold = *x;
     // change the value in x's address to y's value
     *x = *y;
     // change the value in y's address to hold's target
value
     *y = hold;

     return (*x > *y) ? *x : *y;
}
```

Task 7: define a **function** to find and bring back both min and max through input points

```
void max_min( int x, int y, int* pmin, int* pmax);
```

```
void max_min(int x, int y, int* pmin, int* pmax) {
     if (x > y) {
          *pmin = y;
          *pmax = x;
     } else {
          *pmin = x;
          *pmax = y;
     }
}
```

Exercise (not for marks): Create an appropriate declaration for each of the following variables:

a) digits is an array of 10 integers.

```
int digits[10];
```

b) mat is an array of three arrays of five integers.

```
int mat[5][3];
```

c) psa is an array of 10 pointers to char.

```
char* psa[10];
```

d) pstr is a pointer to an array of 10 chars.

```
char (*pstr)[10];
```

Task 8: To identify what are the listed declarations

(1) float *(*p) [4];

p is a pointer to an array containing 4 pointers to floats.

(2) double *p[4][4];

p is a 4 by 4 array of pointers to doubles.

(3) int *(*p[4]) [4];

p is an array of 4 pointers such that each pointer points to an array of 4 pointers to an integer.

(4) void *(p[4]) [4];

p is an array of size 4 such that each element in the array contains an array of void pointers.

Task 9: Identify what are the listed declarations

(1) double (*p) (int, double*) ;

p is a pointer to a function which takes in an int and a pointer to a double and returns a double.

(2) double *(*p) (int, double*);

p is a function pointer in which the function takes in an int and a pointer to a double and returns another pointer to a double.

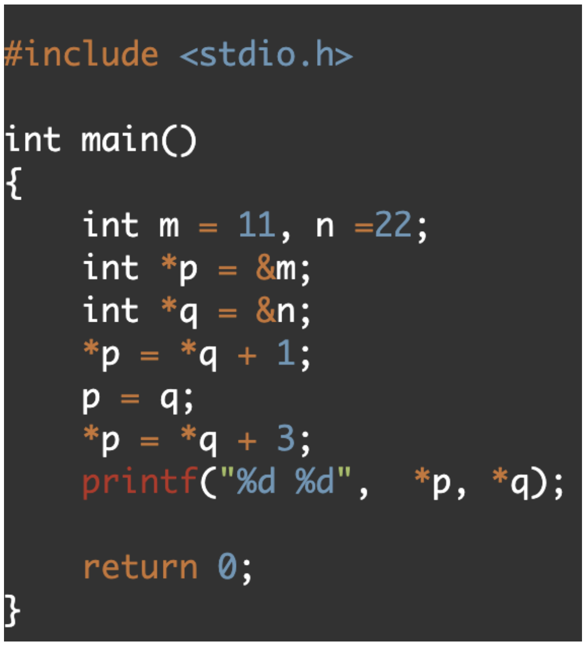
(3) double (*(*p)[3]) (int, double*);

p is a pointer to a size-3 array of function pointers which takes in an int and a pointer to a double, and returns a double.

(4) double (*p[3]) (int, double*);

p is an array of 3 function pointers which takes in an int and a pointer to a double and returns a double.

Task 10: What’s the output for each question

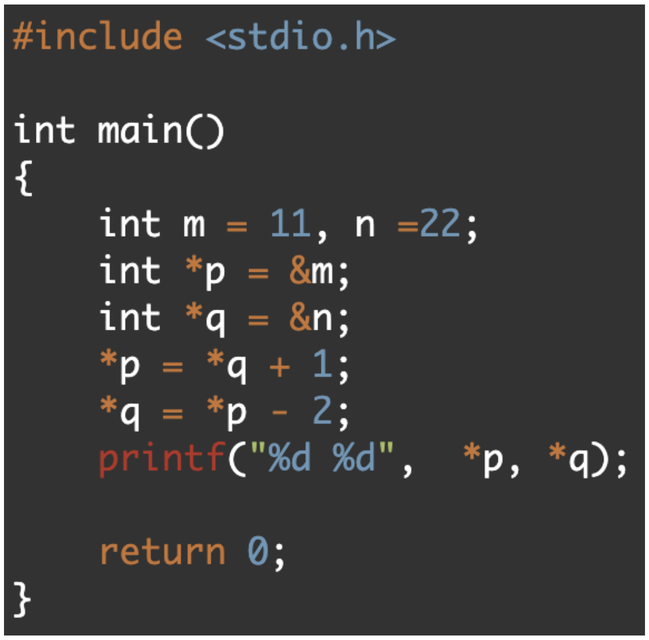


```c
#include <stdio.h>

int main()
{
    int m = 11, n =22;
    int *p = &m;
    int *q = &n;
    *p = *q + 1;
    p = q;
    *p = *q + 3;
    printf("%d %d",  *p, *q);

    return 0;
}
```

```
Your answer:
25 25
```



```c
#include <stdio.h>

int main()
{
    int m = 11, n =22;
    int *p = &m;
    int *q = &n;
    *p = *q + 1;
    *q = *p - 2;
    printf("%d %d",  *p, *q);

    return 0;
}
```

```
Your answer:
```

23 21