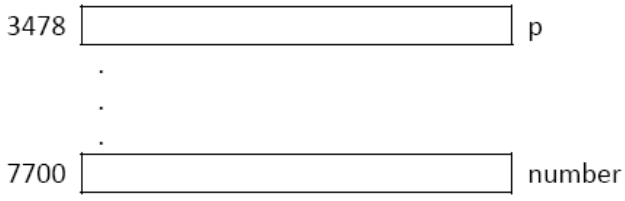
CS100 Tutorial 2 (Fall semester, 2018)

1. Assume the declarations:

```
int number;
int *p;
```

Assume also that the address of the variable `number` is 7700, and the address the pointer variable `p` is 3478. That is:



For each case below, determine the value of

(a) number, (b) &number, (c) p, (d) &p, (e) *p.

All of the results are cumulative.

```
(1)  p = 100; number = 8;
(2)  number = p;
(3)  p = &number;
(4)  *p = 10;
(5)  number = &p;
(6)  p = &p;
```

2. What will be the output of the following program?

```
#include <stdio.h>

void function0(void);
void function1(int h, int k);
void function2(int *h, int *k);

main()
{
   int h, k;
```

```
   h = 5;
   k = 15;
   printf("main 1: \th = %d, k = %d\n", h, k);
   function0();
   printf("main 2: \th = %d, k = %d\n", h, k);
   function1(h, k);
   printf("main 3: \th = %d, k = %d\n", h, k);
   function2(&h, &k);
   printf("main 4: \th = %d, k = %d\n", h, k);

   return 0;
}

void function0()
{
   int h, k;

   h = k = -100;
   printf("function 0.1: \th = %d, k = %d\n", h, k);
}

void function1(int h, int k)
{
   printf("function 1.1: \th = %d, k = %d\n", h, k);
   h = k = 100;
   printf("function 1.2: \th = %d, k = %d\n", h, k);
}

void function2(int *h, int *k)
{
   printf("function 2.1: \th = %d, k = %d\n", *h, *k);
   *h = *k = 200;
   printf("function 2.2: \th = %d, k = %d\n", *h, *k);
}
```

3. Write a function groupDigits() which extracts the digits, that are less than 5 from the non-negative number n, and combines the digits sequentially into a new number. If there is no digit that is less than 5, the new number will be set as –1. The new number is then returned to the calling method. For example, if n = 123456, then the value 1234; and if n = 567, then –1 is returned. Write the function in two versions: function groupDigits1() returns the result, while groupDigits2() passes the result back through the second parameter, nd. You should not use arrays in this problem. The function prototypes are given as follows:

```
   long groupDigits1(long n);
   void groupDigits2(long n, long *nd);
```