UPPSALA UNIVERSITY
DEPARTMENT OF INFORMATION TECHNOLOGY

COMPUTER SYSTEMS/OPERATING SYSTEMS
Fall, 2007

IN-CLASS EXERCISE 2

1. (a) Define a structure called Inventory. It has the following attributes:
   - item number
   - quantity on hand
   - price
   - expiration date (month/year)

   The expiration date must be defined as a struct within the Inventory struct.

   (b) Declare an array of 10 such Inventory items.

   (c) Initialize the expiration date for the 5^{th} inventory item to January 2008.

2. Using the program in Example 8 as a guide, complete the following code segment. The code segment reads two values (an int and a float) from a file “input.txt”. The format of the data in the text file is as shown below:

```
<int>space<float>
```

Code segment:

```
int
main(int argc, char *argv[])
{

  FILE *dataFile;

  int x;
  float y;

  /* Open a file for reading */
  dataFile = fopen("input.txt", _______________);
  if (dataFile == NULL) {
    printf(" Input file does not exist! Quitting...\n");
    exit(1);
  }

  /* Read an integer and a float from the input file */

  ____________________________________________________________

  printf("The values in the file are %d and %f\n", x, y);

  fclose(dataFile);
  return 0;
}
```

3. Given the function prototypes and assuming that function definitions exist, complete the code segment by writing function calls.

```
/* Function: Product
   The function returns the product of the two input parameters.
 */
int Product (int input1, int input2);

/* Function: WriteToFile
   The function writes to the file pointed to by filePtr, the
   values of input1, input2, and output.
 */
void WriteToFile(FILE *filePtr, int input1, int input2,
                 int output1);

int
main(int argc, char *argv[])
{
  FILE *fPtr;

  int inp1 = 3;
  int inp2 = 4;
  int outValue;


  //Call the Product function with inp1 and inp2 as arguments.

  _____________________________________________________________


  //Call the WriteToFile with fPtr, inp1, inp2 and outValue as
  //arguments.

  _____________________________________________________________


  return 0;
}
```

4. For the following code segment (Example 10), use blocks as shown in the slides to explain pointer creation, pointer assignment, and so on. The statements that you must graphically represent have been numbered. Assume j is assigned the address 4000.

```
#include <stdio.h>

int
main(int argc, char *argv[])
{
  int j;
  int *ptr;

  ptr = &j;   /* initialize ptr before using it */   ------ line 1

  *ptr = 4;   /* j <- 4 */                           ------ line 2

  j = *ptr+1; /* j <- ??? */                         ------ line 3
}
```