# CS 167/467

Midterm Exam ANSWERS

August 8, 2006

**Part A**

**1. (5 points)** Describe standard input and standard output.

**Answer:**

Standard input is what the user types in, or the stream of input passed into the program with the shell.

Standard output is what the program prints out for the user to see, or for the shell to redirect.

**2. (5 points)** What value does the function `scanf` return?

**Answer:**

`scanf` returns the number of successfully matched and assigned input items.

**3. (5 points)** What is a pointer?

**Answer:**

A variable that contains a memory address.

**4. (5 points)** Why do we do "`return 0;`" at the end of `main()`?

**Answer:**

To tell the operating system that we are exiting successfully.

**Part B**

Refer to the following code listing for the next few questions:

```
1 #include <stdio.h>
2 #include <string.h>
3 #include <ctype.h>
4
5 #define BIG 1024
6
7 int main(void)
```

```
8  {
9      char line[BIG];
10     int first = 1, i;
11
12     while (1)
13     {
14         if ( fgets(line, sizeof(line), stdin) == 0 )
15             break;
16
17         for (i=0; i<strlen(line); i++)
18         {
19             if (line[i] == ' ' || line[i] == '\n' ||
20                 line[i] == '\t')
21                 first = 1;
22             else
23                 if (first)
24                 {
25                     line[i] = toupper(line[i]);
26                     first = 0;
27                 }
28         }
29
30         printf("%s", line);
31     }
32
33     return 0;
34 }
```

**5. (15 points)** What does this code do?

**Answer:**

It is a filter that capitalizes the first letter of each word.

**6. (10 points)** What happens on line 14?

**Answer:**

The next line of input is read up to a maximum of BIG characters, and we test for EOF.

**7. (10 points)** If `line` contains the string "`hello, world\n`", then what is the value of `i` at line

29?

**Answer:**

The length of the string, or 13

**8. (5 points)** When will this program terminate?

**Answer:**

When the end of the input is reached. Either at the end of the file if a file is redirected to stdin, or when the user presses control-D.

**9. (5 points)** What is the value of `sizeof(line)`?

**Answer:**

1024 (BIG)

**10. (10 points)** What is the maximum possible value of the expression `strlen(line)` on line 17?

**Answer:**

1023 (BIG-1)

Refer to this code listing for the next few questions:

```
1  #include <stdio.h>
2  #include <string.h>
3  #include <ctype.h>
4  #define BIG 1024
5  int main(void)
6  { char line[BIG]; int x = 0, t=1;
7    while (t==1) {
8  int n;
9  t=scanf("%d",&n) ;
10 if(t!= 1){break;}
11 x += n;} printf("%d\n",x);
12           return 0;
13 }
```

**11. (15 points)** What does this code do?

**Answer:**

Sums the numbers read from stdin.

**12. (10 points)** Comment on the style. How would you improve readability of this code? (feel free to mark up the code directly)

**Answer:**

There is much that could be done: indentation, one statement per line, better variable names, comments, vertical spacing.

**13. (10 points)** What purpose does the variable `t` serve? Could we do without it?

**Answer:**

It stores the return value of scanf, we could do without it by testing the return value of scanf directly. In any case, it's not necessary in line 7 (because we break out of the loop with the `break` statement).

Refer to this code listing for the next few questions:

```
1  #include <stdio.h>
2  #include <string.h>
3  int main(void)
4  {
5      char *s = "Lorem ipsum dolor sit amet";
6      char a[42];
7
8      strcpy(a, s);
9
10     if (strlen(s) == sizeof(s))
11         printf("foo\n");
12     if (strlen(a) == sizeof(a))
13         printf("bar\n");
14     if (a == s)
15         printf("baz\n");
16     if (strlen(a) == strlen(s))
17         printf("quux\n");
18
19     s = a+12;
20     a[17] = 'e';
21     a[19] = '\0';
```

```
22      printf("%s\n", s);
23
24      return 0;
25  }
```

**14. (15 points)** What is the output of this program?

**Answer:**

quux
dolores

**15. (15 points)** Explain why each of the `if` conditions is true or false.

**Answer:**

The first is false because `sizeof(s)` is the size of the address, or 4 bytes.

The second is false because `sizeof(s)` is 42, not the length of the string.

The third is false because the array address and the value of the pointer are not the same.

The fourth is true because the length of the two strings is the same.

**16. (15 points)** The following program has several errors and bugs. Mark them and indicate how you would correct them.

```
1   /* This program will calculate the best change to give */
2   void main()
3   {
4       float cost, given, change;
5       int fives, ones, quarters, dimes, nickels, pennies;
6
7       /* read input */
8       printf("What is the total cost? ")
9       scanf("%f",&cost);
10      printf("How much money were you given? ");
11      scanf("%d", given);
12
13      /* calculate change */
14      if (cost > given)
15          printf("That's not enough money.");
```

```
        return 1;

    change = cost-given;
    fives = change / 5;
    change -= fives*5;
    ones = change / 1;
    change -= ones;
    quarters = change / .25;
    change -= quarters * .25;
    dimes = change / .1;
    change -= dimes * .1;
    nickels = change / .05;
    change -= nickels * .05;
    pennies = change / .01;
    change -= pennies;

    /* print answer */
    printf("%f fives, %d ones, %d quarters, %d dimes,
           %d nickels, %d pennies",
           fives, ones, quarters, dimes, nickels, pennies);

    return 0;
};
```

**Answer:**

Here is the fixed code:

```
/* This program will calculate the best change to give */
#include <stdio.h>
int main()
{
    float cost, given, change;
    int fives, ones, quarters, dimes, nickels, pennies;

    /* read input */
    printf("What is the total cost? ");
    scanf("%f",&cost);
    printf("How much money were you given? ");
    scanf("%f", &given);
```

```
13
14      /* calculate change */
15      if (cost > given)
16      {
17          printf("That's not enough money.");
18          return 1;
19      }
20
21      change = given-cost;
22      fives = change / 5;
23      change -= fives*5;
24      ones = change / 1;
25      change -= ones;
26      quarters = change / .25;
27      change -= quarters * .25;
28      dimes = change / .1;
29      change -= dimes * .1;
30      nickels = change / .05;
31      change -= nickels * .05;
32      pennies = change / .01;
33      change -= pennies * .05;
34
35      /* print answer */
36      printf("%d fives, %d ones, %d quarters, %d dimes, "
37             "%d nickels, %d pennies\n",
38             fives, ones, quarters, dimes, nickels, pennies);
39
40      return 0;
41  }
```

**Part C**

**17. (10 points)** Write a function to calculate degrees fahrenheit given degrees celsius ($f = 32 + c\frac{5}{9}$)

**Answer:**

```
1  float fahrentheit(float celsius)
2  {
3      return 32 + celsius * 5.0/9.0;
4  }
```

**18. (5 points)** Give the usage of `scanf` to read the following data: `"387.2 100.8 20 30"` into the variables: `float x, y; int a,b;`.

**Answer:**

```
scanf("%f %f %d %d", &x,&y,&a,&b)
```

**19. (10 points)** Write a program that counts down from 100 to 1.

**Answer:**

```
#include <stdio.h>
int main(void)
{
    int i;
    for (i=100; i>0; i--)
        printf("%d\n",i);

    return 0;
}
```

**20. (5 points)** In computers, images are usually represented as a matrix (or grid) of pixels. For black and white (greyscale) images, pixels are often represented with the `char` integer datatype in C. Declare an array that can hold an 800x600 greyscale image.

**Answer:**

```
char img[800][600];
```

**Part D**

**21. (15 points)** *(CS 467)* What does the following code snippet do?

```
int a[SIZE];
int *p;
/* ... */
while (p<a)
    p += SIZE;
p = a+((p-a)%SIZE);
```

**Answer:**