CSC 111 Fall 2011
Final Examination

Your Name □ **UVicID** □

Instructions

- This final examination consists of 11 pages and 26 questions.
- The questions are worth 2, 4 or 6 points, for a total of 100 points. The points are listed in square brackets at the end of the first line of each question.
- You have 3 hours for this examination. **Time management: approx. 5-7 minutes per question.**
- This final exam is closed-books, closed-notes, no calculators, no gadgets, and no electronic devices.
- Turn in your completed final exam at the front of the class.
- For multiple choice questions, mark all circles that are next to correct choices.
- **Do not leave before 3:30 pm.**

1. The C preprocessor **[2]**

   ○ compiles C programs
   ○ includes text files using #include and substitutes text using #define directives
   ○ builds an application or an executable
   ○ checks for semantic errors

2. C programs are translated into object code using **[2]**

   ○ an editor
   ○ a compiler
   ○ an operating system
   ○ a linker

3. Which one of the following is **NOT** a valid identifier in the programming language C? **[2]**

   ○ hi_There
   ○ 3blindmice
   ○ INT
   ○ first9_

4. How is the following English statement translated into the programming language C? **[2]**

   if j is NOT equal to 0 AND k is greater than equal to 5 then x = 17

   ○ if ((j =! 0) | | (k>=5) {x = 17}

   ○ if ((j != 0) & (k>=5)) {x = 17;}

   ○ if ((j NOT= 0) AND (k > 5)) {x=17;}

   ○ if ((j != 0) && (k>=5)) {x = 17;}

5. What is the output of the following syntactically correct C program? **[4]**

```
#include <stdio.h>

void magic1(int* a, int b) {
   printf ("%d ", *a);    printf ("%d ", b);
   *a = *a * 3 + b;
   printf ("%d ", *a);    printf ("%d ", b);
} /* magic1 */

int main(void) {
   int a = 17, b = 19;
   printf ("%d ", a);    printf ("%d ", b);
   magic1(&a, b);
   printf ("%d ", a);    printf ("%d\n ", b);
   return 0;
} /* main */
```

   ○ 17 19 17 19 70 19 70 19

   ○ 17 19 17 19 51 19 51 19

   ○ 17 19 19 17 70 17 70 17

   ○ 17 19 17 19 70 70 70 19

6. The following syntactically correct C code initializes an array buffer of size 16 with the char '#'. Which for loop below replaces all the array elements with an odd index with the character '@' so that the array contains the following string: "#@#@#@#@#@#@#@#@" **[4]**

```
#define len 16
char buffer[len];
for (k=0; k<len; k++) buffer[k] = '#';
```

   ○ for (k=0; k<len; k++) buffer[k] = '@';

   ○ for (k=0; k<len; k++) buffer[k] = '#@';

   ○ for (k=0; k<len; k++) if (k%2 == 0) buffer[k] = '@';

   ○ for (k=0; k<len; k++) if (k%2 != 0) buffer[k] = '@';

7. Which of the following code fragments correctly defines a structure type "Complex"? **[2]**

○ structure { double re; double im; } Complex;

○ typedef struct {double re, im; } Complex;

○ typedef struct Complex {double re, im }

○ struct Complex (double re, im;);

8. What is the difference between a struct and an array? **[2]**

○ An array groups elements of same data type while a struct groups elements of potentially different data types.

○ Arrays store only integers while structures store integer and string variables.

○ A struct groups elements of the same data type while an array groups elements of different data types.

○ There is no difference between the two.

9. What is the console output of the following syntactically correct C program? **[4]**

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main(void){
        char str[] = "Once upon a time there was a polar bear";
        FILE *ifp;
        FILE *ofp;

        ofp = fopen("vic.txt", "w");
        fputs("I live in Victoria!\n", ofp);
        fclose(ofp);
        ifp = fopen("vic.txt", "r");
        while(fgets(str, strlen(str), ifp) != NULL)  printf("%s", str);
        fclose(ifp);
        return EXIT_SUCCESS;
} /* main */
```

○ Once upon a time there was a polar bear

○ vic.txt

○ I live in Victoria!

○ None of the above

10. Consider the following syntactically correct C declarations and assignments. **[8]**

```
int a;
int d;
int *b;
int *c;
int** e;
a = 17;   b = &a;
c = b;    d = 19;
e = &c;
```

What are the values of the following expressions?

(&a == c)

(b == &d)

(*c == 17)

(**e == *b)

(d == a)

(*b == 19)

(a == b)

(&a == e)

11. What is the output of the following syntactically correct C program? **[4]**

```
#include <stdio.h>
int main(void) {
   int k = 5;
   while (k < 12) {
       printf("%d ", k%7);
       k = k + 1;
   } /* while */
   printf("\n");
   return 0;
} /* main */
```

Output:

12. Consider the following declarations: **[4]**

```
typedef struct {
        int year;
        int month;
        int day;
} Date;
Date dob;
Date *d = &dob;
```

Using variable **d** initialize **dob** with the birthday **July 1, 1867.**

13. What is the output of the following syntactically correct C program? **[4]**

```
#include <stdio.h>
int main(void) {
   int k = 9;
   while (k < 10) {
      printf("%d ", k);
      k = k - 1;
   }
   printf("\n");
   return 0;
} /* main */
```

Output:

14. What is the output of the following syntactically correct C program? **[4]**

```
#include <stdio.h>
int main(void) {
    int k;
    for (k=5;  k>0;  k=k-1) printf("%3d",  k);
    printf("\n");
    return 0;
} /* main */
```

Output:

15. What is the output of the following syntactically correct C program? **[4]**

```
#include <stdio.h>
int main(void) {
    int k;
    for (k=5;  k<6;  k=k-1) printf("%3d",  k);
    printf("\n");
    return 0;
} /* main */
```

Output:

16. Which statement is incorrect? **[2]**

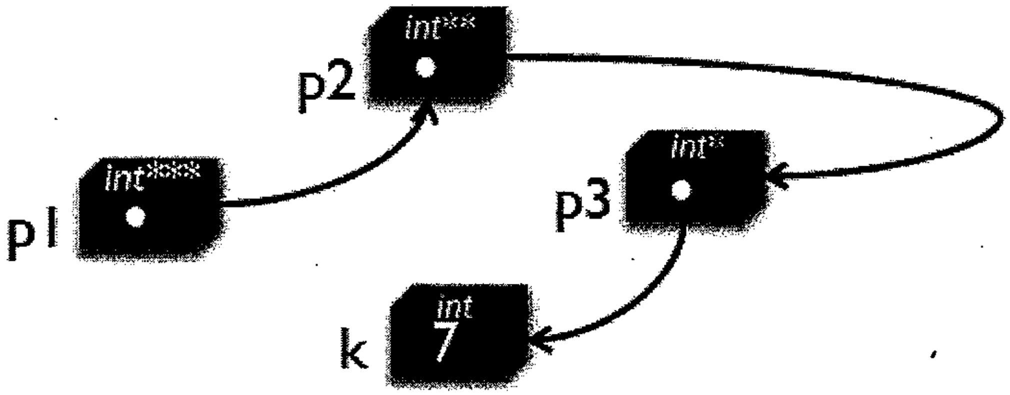
○ A singly linked list can be traversed in one direction.

○ A doubly linked list can be traversed in both directions.

○ It is easier to insert and delete nodes into a doubly linked list than into a singly linked list.

○ A particular node can be found faster in a doubly linked list than in a singly linked list.

17. Realize the following memory configuration using C variable declarations and pointer assignments. Write three assignments using pointers p1, p2, and p3 to store 7 in variable k. **[6]**



18, Which statement is incorrect? **[2]**

- ○ The main operations of a stack are push() and pop().
- ○ The main operations of a queue are enqueue() and dequeue().
- ○ The operations of a stack and a deque are subsets of the operations of doubly linked list.
- ○ A deque provides operations to insert and delete at the same end of the list.

19. What is a binary tree? **[2]**

- ○ A special case of a tree where each node has 0, 1 or 2 children.
- ○ A special case of a tree which stores only binary values (i.e., 0's and 1's).
- ○ A forest consisting of 2 trees.
- ○ A data structure where nodes are linked to other nodes.

20. Write a syntactically correct C function to swap the values of two integer variables. **[6]**

21. Write a syntactically correct C function to shift the elements of a one-dimensional array one position to the right. The last element of the array—the one that drops off the array—is stored in the first element of the array—the one that was freed. **[6]**

```
#include <stdio.h>
void shiftArrayRight(int a[], int len) {
        /* ... Your code goes here */









} /* shiftArrayRight */
```

22. Write a syntactically correct C main() program to open two text files: an input file called **weather.txt** and an output file called **googlemap.txt**. Copy the contents of file **weather.txt** to **googlemap.txt**. Then close the files. **[6]**

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

int main(void) {
        /* ... Your code goes here */









   return EXIT_SUCCESS;
} /* main */
```

23. What data structure can be represented with the following C declarations? **[2]**

```
typedef struct NodeStruct* NodeRef;
typedef struct NodeStruct {
   ItemRef  item;
   NodeRef  prev;
   NodeRef  next;
} Node;
```

○ Singly linked list

○ Doubly linked list

○ Tree

○ Graph

24. Write a syntactically correct C function to convert every lowercase character in the string s to uppercase, and every uppercase character to lowercase (and store the modified character back into s). The standard library functions (defined in ctype.h) 'islower', 'isupper', 'tolower' and 'toupper' may be helpful to implement this function. **[6]**

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

void invertStringCase(char *s){
        /* ... Your code goes here */









} /* invertStringCase */
```

25. How many proper identifiers are in each line according to C syntax? Enter the number for each line in the circle at the front of the line. **[4]**

○ ABC aBc _sysRoutine B777 k 711 int mega %d <stdio.h>

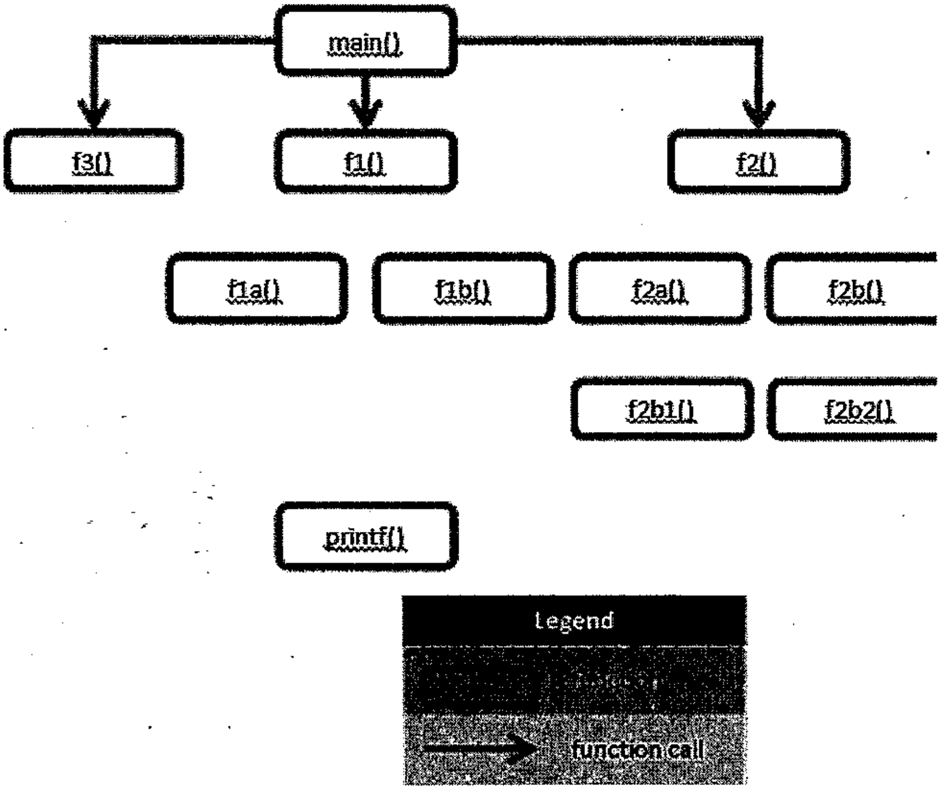
○ This/* is a */an identifier with a caveat #define

○ 17.4 nine 19.6 forty-four hexadecimal k modulo output

○ Check for C++ comments and keywords such as for while long modulo

26. Complete the control flow diagram or call graph for the program below. As the legend indicates rectangles, denote functions and arrows denote call dependencies. The diagram already contains boxes for the 11 functions involved in the program. Thus, you only need to add arrows to the diagram. **[6]**

```
#include <stdio.h>
void f1a(void) {
    printf("f1a\n");
}
void f1b(void) {
    printf("f1b\n");
}
void f2a(void) {
    printf("f2a\n");
}
void f2b1(void) {
     printf("f2b1\n");
}
void f2b2(void) {
     printf("f2b2\n");
}
void f2b(void) {
     f2b1();
     f2b2();
}
void f1(void) {
    f1a();
    f1b();
}
void f2(void) {
     f2a();
     f2b();
}
void f3(void) {
    printf("f3\n");
}
int main(void) {
     f1();
     f2();
     f3();
     return 0 ;
}
```



END