Fall 2022 - MIDTERM SOLUIONS

SECTION 1: Multiple choice (*total of 14 marks: 1 per part*).

1. What is the output from the following code fragment? Assume the line of text entered by the user is: `78 45 xyz`

   (Reminder: When processing `%s`, `scanf` consumes characters up to but not including a space or a newline.)

```
int a = 10, b = 11, c = 12;
char d[ ] = "abcdefg";
printf("Enter a line of text:\n");
a = scanf("%s%d%d", d, &b, &c);
printf("%d %d %d %s", a, b, c, d);
```

   a. 0 abcdefg 10 11
   b. 3 xyz 78 45
   c. 2 abcdefg 78 45
   d. 2 45 12 78
   e. 1 abcdefg 45 11

2. What is the output from the following code fragment?

```
char *a = "Banff";
char b[ ] = "Edmonton";
double c[ ] = {0, 3.141592653589793, 6.283185307179586};
int d[ ] = {sizeof(a), sizeof(b), sizeof(b[2]), sizeof(c)}; int i;
for (i = 0; i < 4; i++) printf("%d ",
  d[i]);
```

   a. 8 8 1 24
   b. 8 9 1 24
   c. 6 9 2 24
   d. 8 9 1 3
   e. 6 9 1 3

3. Which line(s) in the following code fragment will cause a compilation error?

```
int a[4] = { 11, 22, 33 }; // line (1)
const int *b = a;          // line (2)
*b += 1;                   // line (3)
b += 2;                    // line (4)
```

   a. Line (1).
   b. Line (2).
   c. Line (3).
   d. Line (4).

4. What is the output from the following code fragment?

```
char s[ ] = "university";
char *p = s + 3;
int i;
for (i = 0; i < 4; i++) {
  fputc(*p, stdout);
  p++;
}
```

   a. vers
   b. univ
   c. iver
   d. nive
   e. vojw

5. What is the output of the following code segment?

```
char course[] = "XNCM019F2016";
char* sp = &*(course + 1);
while(sp[1] != '2'){
    printf("%c", sp[1]);
    sp = sp + 1 ;
}
```

a. NCM019F
b. NCM019F
c. CM019F
d. This code produces a compilation error
e. None of the above.

6. What is the value of y after this code segment is implemented?

```
long int y;
const char* s= "012345678";
const char* sp = &s[10];
y = s - sp;
```

a. garbage
b. -10
c. 10
d. None of the above.

7. What, if anything is wrong with the following code segment?

Line 1	`char *code = "ASCII";`
Line 2	`char* s;`
Line 3	`s = code;`
Line 4	`code[1] = s[+1];`
Line 5	`s[1] = *code;`

a. There will be a runtime error(s) in this code segment
b. There is a compilation error on the fourth line of this code segment
c. There is a compilation error on the fifth line of the code segment
d. None of the above

8. What is the output of the following C program:

```
#define WINDOWS
int main()
{
    char platform[10];
#ifndef MAC
    strcpy(platform, "LINUX");
#elif !defined(LINUX)
    strcpy(platform,"MAC");
#else
    strcpy(platform, "WINDOWS")
#endif
    printf("%s", platform);
   return 0;
}
```

a. MAC
b. WINDOW
c. LINUX
d. LINUXMAC
e. None of the above

9. Consider the following definition of `struct point` and the next code segment in C:

```
typedef struct Point{double x, y;}Point;
Point x = {200};
Point z = x;
```

What is the value of `z.x` and `z.y`?

a. `z.x` is `0.0` and `z.y` is `200.0`
b. `z.x` is `200.0` and `z.y` is garbage
c. `z.x` is `200.0` and `z.y` is `0.0`
d. None of the above are correct answers.

10. Consider the following C code segment:

```
1  char *p;
2  p = (char*) malloc(20);
3  assert(p != NULL);
4  p += 3;
5  free(p);
```

a. There is an illegal operation on line 2.
b. There is an illegal operation on line 4.
c. **There is an illegal operation on line 5.**
d. There is no illegal operation(s) in this code.

11. What is the output of the following code segment:

```
char s1[] = "ABCDEF";
char* p = s1;
while(*p){
    (*p)++;
    p++;
}
printf("%s\n", s1);
```

Output is:
a. ABCDEF
b. **BCDEFG**
c. FEDCBA
d. None of the above

12. Consider the following C code segment:

```
1  int *p;
2  int n = 5;
3  p = (int*) malloc(n);
4  assert(p != NULL);
5  p[3] = 1000;
```

a. There is an illegal operation on line 3.
b. There is an illegal operation on line 4.
c. **There is an illegal operation on line 5.**
d. There is no illegal operation in this code.

13. What is the output of the following C program?

```
void foo() {
  static int s = 5;
  printf("%d", ++s);
}
```

```
int main() {
  for(int i =0; i < 3; i++)
     foo();

  return 0;

}
```

a. 678
b. 555
c. 567
d. 666
e. None of the above

14. What is the output of the following code segment?

```
char s1[20] = "Barlow";
char s2[]= "Victoria";
const char* p = "ABC";
printf("\n%2d %2d %2d %2d  %2d", (int)sizeof(s1), (int)sizeof(s2),
          (int)sizeof(int), (int)sizeof(p), (int) sizeof(*p));
```

a. 7 9 4 8 4
b. 20 9 4 8 1
c. 20 3 4 8 1
d. 6 8 4 8 1
e. 20 9 4 8 4
f. None of the above

SECTION 2: Functions (*total of 12 marks*).

Part a. *(6 marks.)* Let's define a "square sequence" of numbers as follows: each number except the first is the square of the previous number. So: 2, 4, 16, 256 qualifies, but 2, 4, 15, 225 does not. Given that, write a function definition for `is_square_seq`.

```
int is_square_seq(const int *a, int n);
// REQUIRES: n > 0 and elements a[0] ... a[n-1] exist.
// PROMISES: Return value is 1 if n == 1.
//           Return value is 1 if for each i > 0 and i < n, a[i] is the square
//           of a[i-1].  Otherwise, return value is 0.
```

```
int is_square_seq(const int *a, int n)
{
      int i;

      for (i = 1; i < n; i++)

        if (a[i] != a[i-1] * a[i-1])

          return 0;


      return 1;
}
```

Part b. *(6 marks)* – **Consider the following C struct:**

```
struct object {int i; char t[20];};
```

Write a function definition according to the following function prototype and interface comment. You may *not* call library functions such as `strlen` or `strcmp`.

```
int match_t(const char *s, const struct object *y);
// REQUIRES:s points to a C-string and y points to an object whose member t contains a C string.
// PROMISES:Return value is 1 if the strings found through s and object y match exactly (same
// length, and same sequence of characters).  Otherwise return value is 0.
```

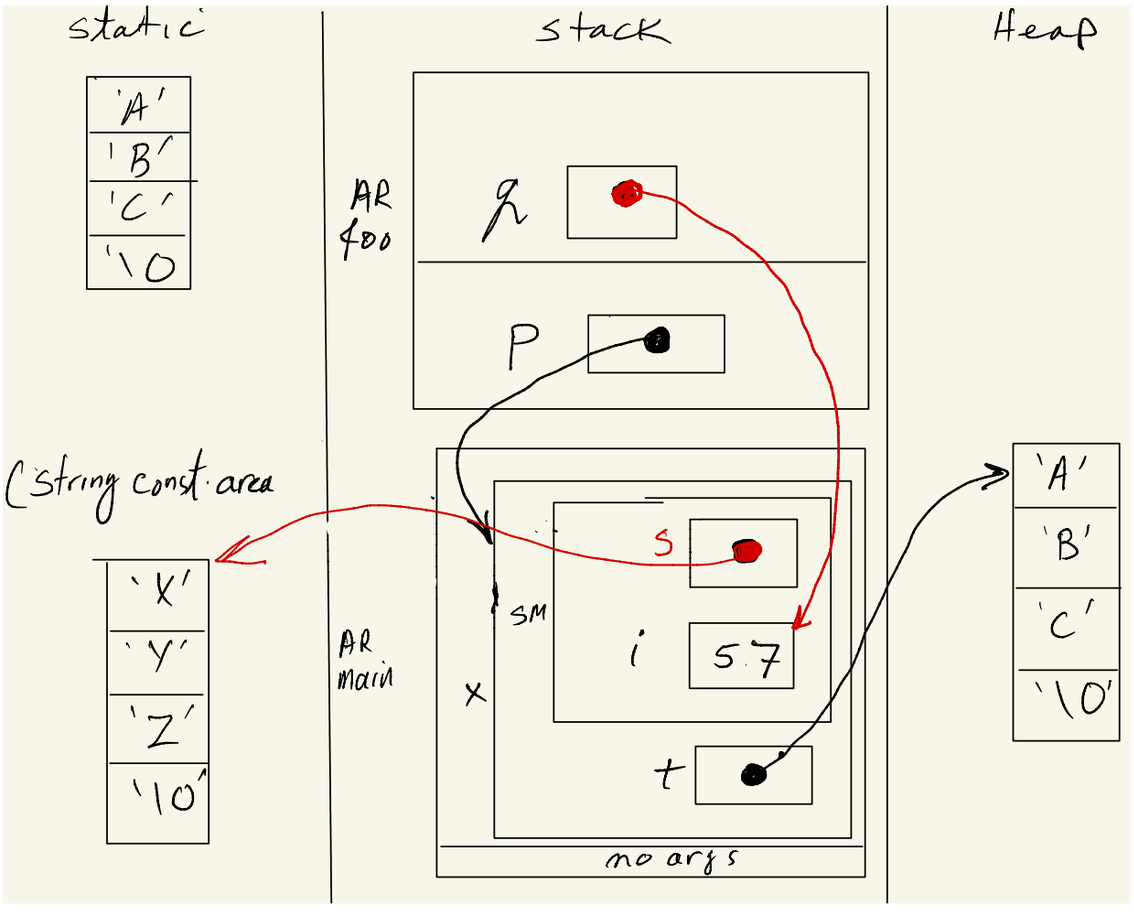
This is one of many possible solutions:

```
int match_t(const char *s, const struct larger *y)
{
     int i = 0;

     while (s[i] != '\0' && s[i] == y->t[i])
          i++;

     if (s[i] == y->t[i])
          return 1; // both chars are '\0'
     else
          return 0; // chars don't match
}
```

SECTION 3: Structure types (*total of 10 marks*).

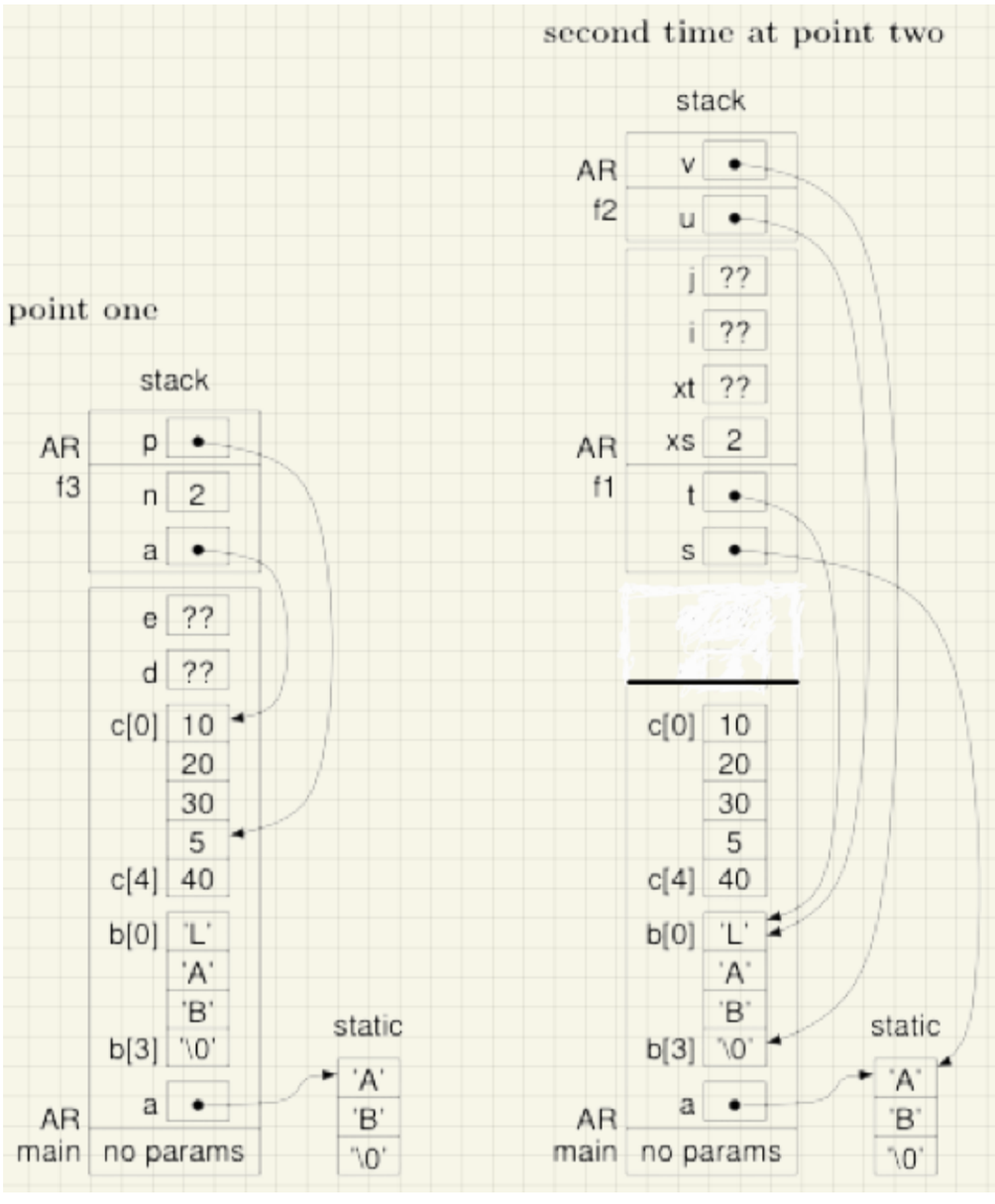
In the space to the right of the program listing, make an AR diagram for point one, in the function foo. Be clear about which items on which memory segments such as stack, static, heap, etc.

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

char global[] = "ABC";

struct smaller {
    const char* s;
    int i;
};

struct larger {
    struct smaller sm;
    char *t;
};

void foo(struct larger *p)
{
    int *q;
    q = &(p->sm.i);
    (*q)++;
    p -> t = malloc(4);
    strcpy(p->t , global);

    // point one
    return;
}

int main(void)
{
    struct larger x = {"XYZ", 56};

    foo(&x);

    return 0;

 }
```



SECTION 4: Arrays, and strings (*12 marks*) - In the following spaces, make memory diagrams for **point-1** and the *second time* the program gets to **point-2**. Be clear about which items are on the stack and which are in static storage.

```
int f2(const char *u){
    const char *v = u;
    while (*v != '\0')
        v++;
    //point-2(when reaches for 2nd time)
    return (v - u);
}

int f1(const char *s, const char *t){
    int xs, xt;
    xs = f2(s);
    xt = f2(t);
    if (xs > xt)
        return 0;
    for (int i = 0, j = xt - xs;
                    i < xs; i++, j++)
    if (s[i] != t[j])
        return 0;
    return 1;
}

int f3(const int *a, int n){
    const int *p = a;
    while (n > 0 && *p >= *a) {
        p++;
        n--;
    }
    // point-1
    return (p - a);
}

int main(void){
    char *a = "AB";
    char b[ ] = "LAB";
    int c[ ] = { 10, 20, 30, 5, 40 };
    f3(c, 5);
    f1(a, b);
    return 0;
}
```



SECTION 5 - Short Answer Questions (*total of 7 marks*)

Part b (5 marks) - The following partial C function, receives two FILE pointers. First pointer is supposed to be for an **input file** and second pointer for an **output file**. The input file is a text-file with several number (any floating number such as 3.4, 0.9999, 345.66, etc) **separated by whitespace**. The function is supposed to read each valid number from input file and write the positive numbers into an output text-file. You can assume that files are successfully opened. It should give an error message if there is any data that is not a valid floating number, and exit the program.

Please read the functions interface comment carefully and complete the missing parts of the function definition.

```
void print_only_positive_numbers(FILE* fp1, FILE* fp2);
// REQUIRES: fp1 refers to a successfully opened input text-file, and fp2 refers to a
// successfully opened output text-file.
// PROMISES: If any of the two FILE pointers are equal to NULL, gives an error message
// terminates the program. Reads all valid number from input text-file and writes ONLY the
// positive numbers into the output text-file. IF there is an invalid data in the file it
// should terminate/exit the program
```

Please complete the missing parts in the following function.

```
void print_only_positive_numbers(FILE* fp1, FILE* fp2){
    int nscan;
    double number;

    while(1){

        nscan = fscanf(fp1, "%lf", &number);

        if(nscan == EOF) break;

        if(nscan == 0) {.
            printf( "An invlid number in the file");
            exit(1);
        }

        if(number >= 0)
            fprintf(fp2, "%lf\n", number);


     } // end of while loop

    fclose(fp1);
    fclose(fp2);
}
```

Part c *(2 marks)*- Complete the blank lines in the following partial definition of the function `my_strlen`, using pointer arithmetic, so that it returns a positive number, which is the length of its argument `str`:

```
int my_strlen(const char* str)
{
     const char* p = str;


     while(*p) {

               p++;

     } // end of while


     return      (int) (p - str);

 }  // end of function
```

SECTION 6: Macros (*total of 4 marks*).

Part a *(2 marks)* - Write a simple macro in C, called RATIO, that receives two numeric arguments, x and y, and returns the ratio of the smaller of the two over the greater one, or returns 1 if they have the same values.k. See the following examples that use macro RATIO:

```
printf("%f", RATIO(5, 10)); // prints 0.500000
printf("%f", RATIO(10, 5)); // prints 0.500000
printf("%f", RATIO(5, 5));  // prints 1.000000
```

Write your macro in the following space :

```
#define RATIO(x, y) (((x)<=(y)) ? (x)/(y) : (y)/(x))
```

Part d. *(2 marks.)* What is the output from the following program? For full credit, you must show how you got your answer.

```
#include <stdio.h>
#define MAC1(x) ((x) > 0)
#define MAC2(y, z) (MAC1(y) - MAC1(z))
int main(void){
  printf("%d\n", MAC2(0, 3));
 return 0;
}
```

Write your answer here:

Answer: **-1**

MAC1(0) ➔ 0
MAC1(3) ➔ 1
0 – 1 = -1