C Assignment 2

Name:
Student ID:

Programming Questions

1. (10 marks) Write three functions `sum1`, `sum2`, `sum3` to sum up the integers between `from` and `to` (`from` and `to` inclusive), using recursion, iteration and algorithm, respectively.

```c
#include <time.h>
#include <stdio.h>

unsigned int sum1(unsigned int from, unsigned int to); /* recursion */
unsigned int sum2(unsigned int from, unsigned int to); /* iteration */
unsigned int sum3(unsigned int from, unsigned int to); /* algorithm */
/* sum(1..n) = n*(n+1)/2 */
/* sum(m..n) = sum(1..n) - sum(1..m-1) */


int main(void)
{
    unsigned int s;
    unsigned from, to;
    double t1, t2;

    printf("Enter the fist integer:");
    scanf("%d", &from);
    printf("Enter the second integer:");
    scanf("%d", &to);

    t1 = clock();
    s = sum1(from, to);
    t2 = clock();
    printf("sum1=%d, %fseconds.\n", s, (t2-t1)/CLOCKS_PER_SEC);

    t1 = clock();
    s = sum2(from, to);
    t2 = clock();
    printf("sum2=%d, %fseconds.\n", s, (t2-t1)/CLOCKS_PER_SEC);

    t1 = clock();
    s = sum3(from, to);
    t2 = clock();
    printf("sum3=%d, %fseconds.\n", s, (t2-t1)/CLOCKS_PER_SEC);

    return 0;
}

unsigned int sum1(unsigned int from, unsigned int to) {
   if (from == to)
          return from;
   unsigned int m = (from+to)/2;
   return sum1(from, m) + sum1(m+1, to);
}
```

```
unsigned int sum2(unsigned int from, unsigned int to) {
   unsigned int i, result = 0;
   for (i = from; i <= to; i++)
          result += i;
   return result;
}

unsigned int sum3(unsigned int from, unsigned int to) {
// sum(1..n) = n*(n+1)/2
// sum(m..n) = sum(1..n) - sum(1..m-1)
  unsigned int result = (to*(to+1))/2;
  if (from > 1)
    result -= (from*(from-1))/2;
  return result;
}
```

2. (5 marks) Write function v_exchange which swaps the values between x[i] and x[SIZE-1-i] (e.g., swap between x[0] and x[9]).

```
#include <stdio.h>
#define SIZE 10

void v_exchange(int a[]);

int main(void)
{ int i, x[SIZE];     /* x[] has 10 int elements */

  for (i=0; i<SIZE; i++)
    x[i] = i;         /* assign i to x[i] */
  v_exchange(x);      /* call for value exchange */

  for (i=0; i<SIZE; i++)
    printf("x[%d]=%d, &x[%d]=%x\n", i, x[i], i, &x[i]);

  return 0;
}

void v_exchange(int a[])    /* this version will not receive full marks */
{
  int i, tmp[SIZE];

  for (i=0; i<SIZE; i++)
    tmp[i] = *(a+i);

  for (i=0; i<SIZE; i++)
    *(a+i) = *(tmp+(SIZE-1)-i);
}

void v_exchange(int a[])
{ int i, tmp;

  for (i=0; i<SIZE/2; i++)
  { tmp = *(a+i);
    *(a+i) = *(a+SIZE-1-i);
    *(a+SIZE-i) = tmp;
  }
}
```

3. (5 marks) Write function `stringcmp` to compare two strings, element by element, for equality. It returns "1" if the two strings are the same and returns "0" if the two strings are different.

```
#include <stdio.h>

int stringcmp( const char *s1, const char *s2 ); /* prototype */

int main( void )
{
  char string1[ 80 ]; /* create a string */
  char string2[ 80 ]; /* create another string */

  printf( "Enter two strings: " );
  scanf( "%s%s", string1 , string2 );
  printf( "The result is %d\n", stringcmp( string1, string2 ) );

   return 0;
}

int stringcmp( const char *s1, const char *s2 )
{
   for ( ; *s1 == *s2; s1++, s2++ )
   {  if ( *s1 == '\0' )
        return 1;
   }
   return 0;
}
```

4. (5 marks) Write function `stringlen` to count the length of a string (number of characters, excluding the '\0' character).

```
#include <stdio.h>

int stringlen( const char *s ); /* prototype */

int main( void )
{
   char string[ 80 ]; /* create char array */

   printf( "Enter a string: ");
   scanf( "%[^\n]", string );
   printf( "%d\n", stringlen( string ) );
   return 0;
}


int stringlen( const char *s )
{
   int x; /* counter */

   /* loop through string */
   for ( x = 0; *s != '\0'; s++ )
     x++;

   return x;
}
```

C Assignment 3

Name:
Student ID:

Exercise Questions

1. See the following two statements.

```
int a[3] = {11, 22, 33};
int *pa = a;
```

Give the values of the following expressions.

```
*a = 11

*(a+2) = 33

*pa = 11

pa[1] = 22
```

2. See the following two statements.

```
int m[4][4] = {{1,3,5,7}, {11,33,55,77}, {2,4,6,8},
{22,44,66,88}};
int (*parr)[4] = m;
```

Give the values of the following expressions.

```
**m = 1

*(*m+2) = 5

*(*(m+1)+1) = 33

*(m[1]+2) = 55

(*(m+2))[3] = 8

(*(parr+3))[2] = 66
```

3. Suppose you are working on a 32-bit machine where the size of an int is FOUR bytes, the size of a char is ONE byte and the size of pointers is FOUR bytes. See the following statements.

```
char *pa[] = {"12", "34", "56"};
int m[2][3] = {{1, 2, 3}, {4, 5, 6}};
int (*ppm)[2][3] = &m;
```

Give the values of the following expressions.

`sizeof(pa)` = 12

`sizeof(*pa)` = 4

`sizeof(**pa)` = 1

`sizeof(ppm)` = 4

`sizeof(*ppm)` = 24

`sizeof(**ppm)` = 12

4. Declare `p`.

`p` is a 5-element array of pointers to `char`.

```
char *p[5];
```

`p` is a pointer to a 10-element `char` array.

```
char (*p)[10];
```

`p` is a function that takes an int argument and returns a pointer to `char`.

```
char *p(int);
```

`p` is a function that takes a char array and returns a pointer to `int`.

```
int *p(char *);
```

`p` is a pointer to a function that takes two `int` arguments and returns a pointer to an `int`.

```
int *(*p)(int, int);
```

`p` is a function that takes no arguments and returns a pointer to a function that takes an `int` argument and returns a pointer to a 10-element `int` array.

```
int (*(*p(void))(int))[10];
```

Programming Questions

5. Complete the following program that prompts user to enter two dates and then indicates which date comes earlier on the calendar.

```
#include <stdio.h>

/* Note: Program assumes years are in the same century. */

struct date {
  int month, day, year;
};

int compare_dates(struct date d1, struct date d2);
void put_date(struct date d);

int main(void)
{
  struct date d1, d2;

  printf("Enter first date (mm/dd/yy): ");
  scanf("%d/%d/%d", &d1.month, &d1.day, &d1.year);
  printf("Enter second date (mm/dd/yy): ");
  scanf("%d/%d/%d", &d2.month, &d2.day, &d2.year);

  if (compare_dates(d1, d2) < 0) {
    put_date(d1);
    printf(" is earlier than ");
    put_date(d2);
    printf("\n");
  } else {
    put_date(d2);
    printf(" is earlier than ");
    put_date(d1);
    printf("\n");
  }

  return 0;
}

int compare_dates(struct date d1, struct date d2)
{
  if (d1.year != d2.year)
    return d1.year < d2.year ? -1 : 1;
  if (d1.month != d2.month)
    return d1.month < d2.month ? -1 : 1;
  if (d1.day != d2.day)
    return d1.day < d2.day ? -1 : 1;

  return 0;
}

void put_date(struct date d)
{
  printf("%d/%d/%.2d", d.month, d.day, d.year);
}
```

6. Rewrite the program in Question 5, using the following function prototype.

```c
   /* loop through rows */
   for ( i = 0; i <= pupils - 1; i++ ) {

      /* loop through columns */
      for ( j = 0; j <= tests - 1; j++ ) {

         /* if current grade is higher than highGrade */
         if ( grades[ i ][ j ] > highGrade ) {
            highGrade = grades[ i ][ j ];
         } /* end if */

      } /* end for */

   } /* end for */

   printf( "\n\tThe highest grade is %d\n", highGrade );
} /* end function maximum */

/* calculate average */
void average( int grades[][ EXAMS ], int pupils, int tests )
{
   int i; /* loop counter */
   int j; /* loop counter */
   int total; /* sum of all grades */

   printf( "\n" );

   /* loop through rows */
   for ( i = 0; i <= pupils - 1; i++ ) {
      total = 0;

      /* loop through columns */
      for ( j = 0; j <= tests - 1; j++ ) {
         total += grades[ i ][ j ];
      } /* end for */

      printf( "\tThe average for student %d is %.1f\n",
             i + 1, ( double ) total / tests );
   } /* end for */

} /* end function average */

/* print the contents of the array */
void printArray( int grades[][ EXAMS ], int pupils, int tests )
{
   int i; /* loop counter */
   int j; /* loop counter */

   printf( "\n\t                  [ 0 ]  [ 1 ]  [ 2 ]  [ 3 ]" );

   /* loop through rows */
   for ( i = 0; i <= pupils - 1; i++ ) {
      printf( "\n\tstudentGrades[ %d ] ", i );

      /* loop through columns */
      for ( j = 0; j <= tests - 1; j++ ) {
         printf( "%-7d", grades[ i ][ j ] );
      } /* end for */
```

```
int compare_dates(struct date *, struct date *);
void put_date(struct date *);
```

7. Write function `swap_ptr` which swaps the values between `ptrp` and `ptrq`. You also need to provide the function prototype for `swap_ptr`.

```
void swap_ptr(int **, int **);

int main(void)
{ int p = 11, q = 22;
  int *ptrp = &p, *ptrq = &q;
  int **ppp = &ptrp, **ppq  = &ptrq;
  swap_ptr(ppp,ppq);        /* &ptrp, &ptrq passed */
                                  /* to swap_ptr() */
  return 0;
}

void swap_ptr(int **ppx, int **ppy)
{ int *tmp;
  tmp = *ppx;
  *ppx = *ppy; /* the values stored at */
  *ppy = tmp;  /* &ptrp, &ptrp get swapped */
}
```

8. Complete the following program. The program should provide user a text menu with 5 options (see below). Options 0~3 are implemented by functions. That is, making choice of these four options will call corresponding functions.

```
Enter a choice:
0 Print the array of grades
1 Find the minimum grade
2 Find the maximum grade
3 Print the average on all tests for each student
4 End program

#include <stdio.h>
#define STUDENTS 3
#define EXAMS 4

/* function prototypes */
void minimum( int grades[][ EXAMS ], int pupils, int tests );
void maximum( int grades[][ EXAMS ], int pupils, int tests );
void average( int grades[][ EXAMS ], int pupils, int tests );
void printArray( int grades[][ EXAMS ], int pupils, int tests );
void printMenu( void );

int main(void)
{

   /* pointer to a function that takes as parameters a
      two-dimensional array and two integer values */
   void ( *processGrades[ 4 ] )( int [][ EXAMS ], int, int )
                      = { printArray, minimum, maximum, average};

   int choice = 0; /* menu choice */
```

```c
   /* array of student grades */
   int studentGrades[ STUDENTS ][ EXAMS ] = { { 77, 68, 86, 73 },
                                              { 96, 87, 89, 78 },
                                              { 70, 90, 86, 81 } };

   /* loop while user does not choose option 4 */
   while ( choice != 4 ) {

      /* display menu and read user's choice */
      do {
         printMenu();
         scanf( "%d", &choice );
      } while ( choice < 0 || choice > 4 ); /* end do...while */

      /* pass choice into the array */
      if ( choice != 4 ) {
         ( *processGrades[ choice ] )( studentGrades, STUDENTS, EXAMS );
      } /* end if */
      else {
         printf( "Program Ended.\n" );
      } /* end else */

   } /* end while */

   return 0; /* indicate successful termination */
} /* end main */

/* search for the minimum value */
void minimum( int grades[][ EXAMS ], int pupils, int tests )
{
   int i; /* loop counter */
   int j; /* loop counter */
   int lowGrade = 100; /* set lowGrade to highest possible score */

   /* loop through rows */
   for ( i = 0; i <= pupils - 1; i++ ) {

      /* loop through columns */
      for ( j = 0; j <= tests - 1; j++ ) {

         /* if current grade is lower than lowGrade */
         if ( grades[ i ][ j ] < lowGrade ) {
            lowGrade = grades[ i ][ j ];
         } /* end if */

      } /* end for */

   } /* end for */

   printf( "\n\tThe lowest grade is %d\n", lowGrade );
} /* end function minimum */

/* search for maximum value */
void maximum( int grades[][ EXAMS ], int pupils, int tests )
{
   int i; /* loop counter */
   int j; /* loop counter */
   int highGrade = 0; /* set highGrade to lowest possible score */
   } /* end for */

   printf( "\n" );
} /* end function printArray */

/* display the menu */
void printMenu( void )
{
   printf( "\n\tEnter a choice:\n"
          "\t  0  Print the array of grades\n"
          "\t  1  Find the minimum grade\n"
          "\t  2  Find the maximum grade\n"
          "\t  3  Print the average on all"
          " tests for each student\n"
          "\t  4  End program\n"
          "\t? " );
} /* end function printMenu */
```