NATIONAL UNIVERSITY OF SINGAPORE

CS1010 – PROGRAMMING METHODOLOGY

(Semester 2: AY2015/16)

Time Allowed: 2 Hours

INSTRUCTIONS TO STUDENTS

1. This assessment paper consists of **THIRTEEN** questions and comprises **TEN** printed pages.
2. This is an **OPEN BOOK** assessment.
3. Calculators and electronic dictionaries are not allowed.
4. Answer all questions, and write your answers in the **ANSWER SHEETS** provided.
5. Fill in your Student Number with a pen, clearly on every odd-numbered page of your ANSWER SHEETS.
6. You may use **2B pencil** to write your programs.
7. Note that penalty will be given for programs that are unclear or unnecessarily long.
8. You must submit only the ANSWER SHEETS and no other document.

Section A: Multiple Choice Questions (MCQs) [12 marks]

There are six MCQs in this section. Each MCQ has one correct answer and is worth 2 marks. There is no penalty for wrong answer.

Q1. Which of the following statements is true?

A. When declaring a 2D array, the number of columns of the array can be omitted.
B. The sizes of the rows in a 2D array can be different.
C. If one of the parameters of a function is a 2D array, the function can be called with a 2D array with any number of rows.
D. Given a 2D array **arr**, its first column can be referred to by the expression **arr[][0]**.
E. None of the above options is true.

Q2. What is the output of this code fragment? (The integer value of 'A' is 65.)

```
char D = 'H', H = 'D';
printf("%d\n", D+(D-H)/2);
```

A. 66
B. 68
C. 70
D. 72
E. 74

Q3. What is the value of **b** at the end of the following code fragment?

```
double a = rand() / RAND_MAX + rand() % 2
int b = (int)(a * 2) * 2 + 3 ;
```

A. A random integer 3 or 7
B. A random integer 3, 7 or 11
C. A random integer in [3, 11]
D. A random odd integer in [3, 11)
E. A random odd integer in [3, 11]

Q4. Given the following structure type definition and function definitions,

```
typedef struct {
    char name[50];
    int age;
} student_t;

void func1(char name[], int age){
    strcat(name, "*");
    age += 5;
}

void func2(student_t stu[]){
    int i;
    for (i = 0; i < 3; i++){
        stu[i].age/=2;
    }
}

void func3(student_t *ptr){
    strcpy(ptr->name, "Rose");
    ptr->age += 10;
}
```

what is the output of this program fragment?

```
int i;
student_t stu[3] = {{"John",25}, {"Mary",20}, {"Otto",33}};

func1(stu[1].name, stu[1].age);
func2(stu);
func3(stu);

for (i = 0; i < 3; i++)
     printf("%s %d ", stu[i].name, stu[i].age);
```

A. Rose 22 Mary* 10 Otto 16
B. Rose 35 Mary* 20 Otto 33
C. John 12 Mary 10 Otto 16
D. John 12 Mary* 12 Otto 16
E. None of the above options is true.

Q5. In the process of computing **f(9)**, how many times is **f()** called (excluding **f(9)** itself)?

```
int f(int n){
    if (n <= 1) return n;
    else if (n % 2 == 0) return 2 * f(n/2);
    else return f(n-1) + f(n-3);
}
```

A. 6
B. 7
C. 8
D. 9
E. 10

Q6. How many comparisons (between the key and an element in the list) does it take to search for 30 in the array {1, 6, 10, 11, 14, 20, 22} using the linear search and binary search as introduced in the lecture notes, respectively?

A. 6 and 3
B. 7 and 3
C. 7 and 4
D. 8 and 3
E. 8 and 4

Section B: Short Structured Questions [14 marks]

Q7. Write the output of the following program. [5 marks]

```
int main(void){
    int a = 5, b = 0, c = f(&b, a);
    printf("%d %d %d\n", a, b, c);
    return 0;
}

int f(int *a, int b){
    int sum = b;
    while (*a < b){
        sum += b--;
        ++(*a);
    }
    return sum;
}
```

Q8. Mark the positions of 1s in the 2D array **arr** at Point (a) and Point (b). [4 marks]

```
int arr[5][5] = {{0}};
int i, j, k, temp;

for (i = 0; i<5; i++){
    arr[0][i]=1;
    arr[i][2]=1;
}

// Point (a)

for (i = 0; i < 5; i++){
    for (j = 0; j < 5; j++){
        if (i+j < 4){
            temp = arr[i][j];
            arr[i][j] = arr[4-j][4-i];
            arr[4-j][4-i] = temp;
        }
    }

// Point (b)
```

Q9. Complete the following function

void copynword(char src[], char dest[], int n)

which copies the first **n** (**n** >=1) word(s) from **src** to **dest**. (If **n** is greater than the number of words in **src**, all words in **src** are copied.)

For example, if **src** contains the string "Be the change you wish to see in the world", after calling **copynwords(src, dest, 3)**, **dest** should contain the string "Be the change".

You may assume that 1) the given string consists of only letters and spaces, 2) there is at least one word in the given string, 3) all words are separated by exactly one space, and 4) the size of **dest** is sufficient for storing the words to be copied. [5 marks]

```
void copynwords(char src[], char dest[], int n){
    dest[0] = '\0';
    char *p = strtok(src, " ");
    while (p != NULL && n > 0)
       [   (a)   ];
       strcat(dest, " ");
       p = [   (b)   ];
       n--;
    }
    dest[[   (c)   ]] = '\0';
}
```

Section C: Problem Solving [54 marks]

Q10. Work records [24 marks]

The following structures are used to store information about the work records of the employees in a company.

```
typedef struct {
   int day;              // Day of the work record [1,31]
   int start;            // Start time [0,24]
   int end;              // End time [0,24], end >= start
} record_t;

typedef struct {
   char name[81];         // Name of the employee
   int numRecords;        // Number of work records [1,200]
   record_t records[200];// Work records
} employee_t;
```

(a) Write a function

```
int readRecords(employee_t employees[])
```

to read data from a text file called **records.in** into an **employee_t** array called **employees** and return the number of employees read. The first line of the file contains the number of employees. This is followed by the information of each employee, which consists of the name of the employee, the number of work records, and the work records of this employee.

You should check that the file can be opened properly and handle the error when the opening of the file fails.

An example of the file **records.in** is as shown on the right. There are 3 employees in the file. The first employee is John, who has come to work on 4 days: day 4 (20-22), day 5 (9-17), day 6 (10-17) and day 7 (9-12).

You may assume that 1) the number of employees in the company is in [1, 100], 2) the name of an employee does not contain whitespace characters, and 3) the numbers are within the specified ranges.

[8 marks]

```
3
John 4
4 20 22
5 9 17
6 10 17
7 9 12
Mary 2
4 1 23
5 6 12
Sally 2
5 12 14
6 8 15
```

Q10. *(continue…)*

(b) Write a function

```
void sortEmployees(employee_t employees[], int num)
```

that sorts a list of employees by the total number of hours they have worked **in descending order**. If there are multiple employees who have worked the same number of hours, the order among these employees does not matter.

For example, if the list of employees is as given in the sample text file in part (a), the sorted list should be in the order of Mary (22 + 6 = 28 hours), John (2 + 8 + 7 + 3 = 20 hours) and then Sally (2 + 7 = 9 hours).

Your function should take in an **employee_t** array called **employees**, which represents the list of the employees to be sorted, as well as an integer **num**, which represents the number of employees in the list. [8 marks]

(c) Write a function

```
int lonelyEmployee(employee_t employees[], int num,
                   char name[])
```

that returns 1 if there exists a day on which there is only one employee who has come to work (otherwise 0). If such a day exists, the function also returns the name of the employee who has come to work on that day.

For example, if the list of employees is as given in the sample text file in part (a), the function should return 1 and the string "John" since he is the only one who has come to work on day 7.

The first two inputs of this function are the same as the ones in part (b). In addition, this function takes in a char array **name** for returning the name of the employee.

You may assume that the size of **name** is sufficient for storing the name of the employee. If there is more than one such employee, the function may return the name of any of them. [8 marks]

Q11. Movie Taste

A local movie company has conducted a survey with **m** participants. Each participant was asked to give a rating (1~5 stars) to each of the **n** movies.

The results of the survey are stored in a 2D **int** array **ratings** with each row containing the ratings from one participant and each column containing the ratings for one movie.

	[0]	[1]	[2]	[3]	[4]
[0]	4	4	3	3	2
[1]	4	3	1	1	5
[2]	5	5	5	2	2
[3]	4	3	1	1	4

For example, the 2D array on the right shows the ratings of 4 participants on 5 movies. Participant 0 gives a rating of 4 stars for movie 0, while participant 3 gives a rating of 1 star for movie 2.

As a token of appreciation, the company has decided to give away a pair of tickets to the two participants with the most similar movie taste.

To keep things simple, the company uses **Euclidean distance** as the metric for similarity between two participants *i* and *j* and the formula is as shown below:

$$distance = \sqrt{(r_{i1} - r_{j1})^2 + (r_{i2} - r_{j2})^2 + \ \dots + (r_{in} - r_{jn})^2}$$

where $(r_{i1}, r_{i2}, \dots, r_{in})$ and $(r_{j1}, r_{j2}, \dots, r_{jn})$ are the ratings given by the two participants respectively on the **n** movies.

The two participants with the smallest Euclidean distance win the pair of tickets.

For example, in the 2D array given above, the Euclidean distance of participant 1 and participant 3 is $\sqrt{(4-4)^2 + (3-3)^2 + (1-1)^2 + (1-1)^2 + (5-4)^2} = 1$. This is the smallest among all pairs of participants and hence they win the pair of tickets.

Write a function **`mostSimilar()`** which takes in a 2D array of ratings, an integer representing the number of users, and another integer representing the number of movies. This function should return the indices of the two participants with the most similar movie taste. If there is more than one pair of participants with the smallest Euclidean distance, it does not matter which pair the function returns.

You are to determine the appropriate parameters and return type for this function. Make sure that you choose meaningful names for the parameters. You may assume that there are at least 2 participants and the number of movies is in [1, 100]. [10 marks]

Q12. Lock Screen

A tech company is developing a lock screen app for handphones.

This app works by first asking the user to set a frequency pattern in the form of

$$freq_1letter_1freq_2letter_2...freq_nletter_n$$

where each *freq* is a single digit representing a frequency, and each *letter* is an uppercase letter in the English alphabet (e.g., "3E4A1K", "5B2C", "9Z9Y9X").

Once a pattern is set, the handphone can only be unlocked by entering a sequence of uppercase letters which matches the letter frequencies specified in the pattern.

For example, if the pattern is "3E4A1K", then the handphone can be unlocked by any sequence which contains **exactly** three Es, four As and one K (e.g., "AEEEKAAA" and "AEAEAEAK"). In contrast, sequences such as "AAAEEEK" and "CAAAAEEEK" do not work since the former contains only three As while the latter has one extra C.

Write a function

```
int match(char str[], char pat[])
```

to check whether a given string **str** matches with a given pattern **pat**. This function returns 1 if so or 0 otherwise. You may assume that **str** contains only upper case letters while **pat** contains only digits and uppercase letters in the specified format. [12 marks]

Q13. Coupon

Madam Tan is a thrifty housewife who loves to collect coupons and use them in all her purchases. She has recently collected a set of coupons, each of which allows her to buy a certain number of an item at a certain price (e.g., buy 3 boxes of chocolates at the price of 10 dollars). Each coupon can only be used at most once and cannot be used partially (i.e., the number of items purchased using a coupon must be the same as number specified on the coupon).

Now she wants to know what the minimum cost is for buying exactly **n** units of an item using these coupons only.

For example, let say she has 4 coupons: "Buy 3 at 10 dollars", "Buy 2 at 4 dollars", "Buy 2 at 4 dollars" and "Buy 1 at 3 dollars", and she wants to buy 4 units of the item. The minimum cost is then 8 dollars (using the second and the third coupon).

Write a recursive function

```
int minimumCost(coupon_t coupons[],
                int numCoupons, int units)
```

to help Madam Tan compute the minimum cost.

This function should take in a **coupon_t** array called **coupons**, which represents the coupons, as well as an integer **numCoupons**, which represents the number of coupons in the list. This function also takes in another integer **units**, which represents the number of units to be purchased.

The structure type definition for **coupon_t** is as shown below:

```
typedef struct {
   int quantity;        // Number of items purchasable
                        // with this coupon
   int price;           // Total price to be paid for
                        // using this coupon
} coupon_t;
```

This function returns the minimum cost if it is possible to make the purchase using these coupons, or -1 otherwise.

No marks will be given if the function is not recursive. [8 marks]

=== END OF PAPER ===