PART A – OBJECTIVE QUESTIONS **[20 Marks]**

Part A consists of 20 Objective Questions. Choose the correct answer, and write your answer in the answer booklet. Each question carries 1 mark.

1. Which of the following statements describe a data structure?

   A. A specialized format for organizing, processing, retrieving and storing data.
   B. A step-by-step instruction to perform a task within a finite period.
   C. A systematic approach using engineering principle in software development.
   D. A mathematical expression to solve a problem.

2. Which of the following is a non-linear data structure?

   A. Stack
   B. Array
   C. Graph
   D. Queue

3. Azizah needs to develop a software for UTM Application System. Currently, she is in the process of monitoring system performance. She identifies the errors which are not detected during the system testing and enhancement to the system. In software development process, this phase is called as _____.

   A. Design
   B. Testing
   C. Analysis
   D. Maintenance

4.

```
1  :       :       :
2  int main()
3  {
4    ifstream inp("inp.txt");
5    string str;
6
7    //???
8    inp >> str;
9    :       :       :
```

Given the above code segment and the following inputs in a file: "***Data Structure and Algorithm***". What statement can be used (in **line 7**) to move the cursor to the correct position if you want to get the last word ("***Algorithm***") from the input?

A. `inp.seekg(-9, ios::end);`
B. `inp.seekp(-9, ios::end);`
C. `inp.seekg(19, ios::beg);`
D. `inp.seekp(19, ios::beg);`

5.

```
1  :      :      :
2  int main()
3  {
4    ifstream inp("input.txt");
5    string str;
6
7    inp >> str;
8    cout << inp.tellg();
9    :      :      :
```

What is the output of the above code segment, if the input file contains the following sentences: "***People say nothing is impossible, but I do nothing every day.***"?

A. 62
B. 7
C. 6
D. 1

6.

```
1   //Program 1
2   :      :      :
3   struct Student {
4     char name[50];
5     int age;
6   };
7
8   int main()
9   {
10    Student me = {"Rahim", 19};
11    Student group1[3]= {"Aini", 20, "Jack", 19, "Mary", 20};
12    char gender = 'M';
13    double x = 34.12;
14    ofstream file;
15
16    file.open("out.dat", ios::bin | ios::out);
17    file.write(gender, 1);
18    file.write((char *)&x, sizeof(double));
19    file.write((char *)me, sizeof(Student));
20    file.write((char *)group1, sizeof(group1));
21    :      :      :
```

Which of the following statements causes a syntax error in the above program segment (**Program 1**)?

i. `file.write(gender, 1);`
ii. `file.write((char *)&x, sizeof(double));`
iii. `file.write((char *)me, sizeof(Student));`
iv. `file.open("out.dat", ios::bin | ios::out);`
v. `file.write((char *)group1, sizeof(group1));`

A. ii, iv and v
B. i, ii, and iii
C. i, iii, and v
D. i, iii, and iv

7. Choose the **TRUE** statements for recursive function in **Figure 1**?

```
1  int compute (int d) {
2    if (d == 0)
3       return 2;
4    else
5       return (5 * compute(d-2));
6  }
```

Figure 1: Recursive function

i. There is no base case defined to stop the recursive calls.
ii. There is no general case defined to call the recursive function.
iii. The program will not or never return 2 if **`d`** is odd number.
iv. The program will return 10 when **`compute(2)`** is called and executed.

A. i and ii
B. iii and iv
C. i and iii
D. All above is true

8.

```
1  void decNum(int a) {
2    if (a==1)
3      cout << "Day";
4    else {
5      cout << a;
6      decNum(a-1);
7      cout << a;
8    }
9  }
```

Figure 2: Recursive function

What is the output of the function in **Figure 2**, if the statement: **`decNum(3);`** is executed?

A. `3Day2`
B. `33Day22`
C. `23Day32`
D. `32Day23`

9. Which of the following statements is **TRUE**?

A. Recursion uses loop control structure such as **`for`**, **`while`** and **`do..while`** to do repetition.
B. Iteration cannot be infinitely repeated like recursion.
C. Recursion consumes more memory than iteration.
D. Iteration is always better and easier than recursion

10. Which of the following growth-rate functions increases very fast once the problem size increase?

    A. 1
    B. n
    C. 2^n
    D. $\log_2 n$

11. The **big-O notation** for an algorithm with total steps of $\mathbf{3 + 8n + 4n^2 + 7 \log_2 n}$ is:

    A. $4n^2$
    B. $n^2 + \log_2 n$
    C. n^2
    D. $\log_2 n$

12. Answer the question based on the following code segment:

```
1  int n;
2  int count = 1;
3  int i = 1;
4
5  while (i < n) {
6    count++;
7    i = i * 2;
8  }
```

    The number of steps for the above algorithm is:

    A. $3 + 3 (1 + \log_2 n)$
    B. $6 + 3n \log_2 n$
    C. $3 + 5n \log_2 n$
    D. $\log_2 n$

13.

```
1  bool needNextPass = true;
2  for (int k = 1; k < listSize && needNextPass; k++) {
3      needNextPass = false;
4      for (int i = 0; i < listSize - k; i++)
5         if (list[i] > list[i + 1]) {
6            //swap [list[i] with list[i+1];
7            needNextPass = true;
8         }
9  }
```

    Based on the algorithm above, what type of the sorting algorithm that it used?

    A. Bubble sort algorithm
    B. Improved bubble sort algorithm
    C. Selection sort algorithm
    D. Insertion sort algorithm

14. If the five-element array **[30, 28, 12, 24, 8]** is sorted in ascending order using selection sort, what is the order of the array after **pass = 2**?

    A. [30, 28, 12, 24, 8]
    B. [8, 28, 12, 24, 30]
    C. [8, 24, 12, 28, 30]
    D. [8, 12, 28, 24, 30]

15. In a bubble sort on an array of six elements **[2, 9, 5, 4, 8, 1]**, what is the order of the array when executing the pass 1 according to ascending order?

    A. 2 4 5 1 8 9
    B. 2 4 1 5 8 9
    C. 2 1 4 5 8 9
    D. 2 5 4 8 1 9

16. For the worst case analysis, which of the following simple sorting algorithm will have **O(n)** complexity?

    A. Quick Sort
    B. Insertion Sort
    C. Bubble Sort
    D. Selection Sort

17. Which of the following is **NOT TRUE** about merge sort?

    A. Worse case and average case for merge sort is $O(n \log_2 n)$.
    B. Performance of merge sort will be affected by the initial order of the array items.
    C. Merge sort is an extremely fast algorithm.
    D. Merge sort requires a second array as large as the original array.

18. Which of the following is the correct code for the merge sort?

    A.

```
void mergeSort(int myArray[], int first, int last)
{
    if (first < last)
    {
        int mid = (first + last) / 2;
        mergesort(myArray, first, mid);
        mergesort(myArray, mid + 1, last);
        merge(myArray, first, mid, last);
    } // end if
} // end mergesort
```

B.

```
void mergeSort(int myArray[], int first, int last)
{
    if (first < last)
    {
        int mid = (last - first) / 2;
        mergesort(myArray, first, mid);
        mergesort(myArray, mid + 1, last);
        merge(myArray, first, mid, last);
    } // end if
} // end mergesort
```

C.

```
void mergeSort(int myArray[], int first, int last)
{
    if (first < last)
    {
        int mid = (last - first) / 2;
        merge(myArray, first, mid, last);
        mergesort(myArray, first, mid);
        mergesort(myArray, mid + 1, last);
    } // end if
} // end mergesort
```

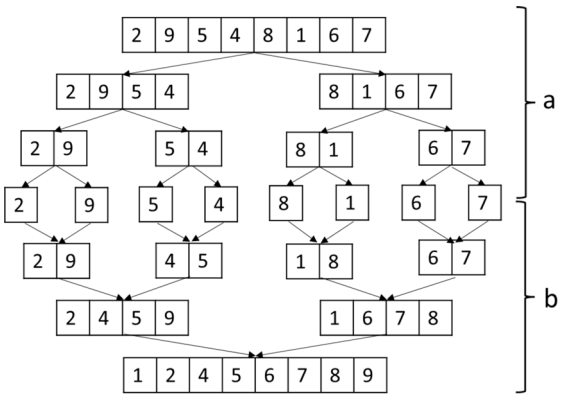
D.

```
void mergeSort(int myArray[], int first, int last)
{
    if (first < last)
    {
        int mid = (first + last) / 2;
        merge(myArray, first, mid, last);
        mergesort(myArray, first, mid);
        mergesort(myArray, mid + 1, last);
    } // end if
} // end mergesort
```

19. Which of the following is **NOT TRUE** about quick sort?

A. The efficiency of quick sort depends on the initial order of the array items.
B. Pivot can be chosen at random, first element, or from the last element in the array.
C. The worst case for quick sort occurs when the smallest item or the largest item always be chosen as pivot.
D. Quick sort is fast in practice.

20.



In merge sort of an array of eight elements, what is the term of a & b in the above figure?

A. a = divide, b = conquer
B. a = divide, b = compare
C. a = swap, b = conquer
D. a = swap, b = compare

PART B - STRUCTURED QUESTIONS **[80 Marks]**

Part B consists of 5 structured questions. Answer all questions in the answer booklet. The marks for each part of the question is as indicated.

Question 1 **[10 Marks]**

Given **Program B1** that contains a few errors. Identify which code lines contain the errors and write the correct code that can save and read **`double`** array in a binary file. This program should display the output as in **Figure 3**.

Line	Codes Program B1
1.	`#include <iostream>`
2.	`#include <fstream>`
3.	`using namespace std;`
4.	
5.	`int main() {`
6.	`  int x, n = 5;`
7.	`  double score[n] = {80.5, 56.2, 100.42, 75.30,30.6};`
8.	
9.	`    stream out("scores", ios:out || ios:binary);`
10.	`    cout.write((char *) score, sizeof(double));`
11.	`    out_close;`
12.	
13.	`    stream in("scores", ios:in || ios:binary);`
14.	`    cin.read((char *) score, sizeof (double));`
15.	
16.	`    //To display how many bytes have been read`
17.	`    cout << seekg() << " bytes read\n";`
18.	`    for(int i=0; i<n; i++) //To show values read from the input file`
19.	`      cout << score << " ";`
20.	
21.	`    in_close;`
22.	`    return 0;`
23.	`} //main`

```
40 bytes read
80.5 56.2 100.42 75.3 30.6
```

Figure 3: Program B1 Output

Line	Correct code

Question 2 **[15 Marks]**

Given the following program, answer the given questions:

```
1  int find(int a, int b) {
2    if (a == 0)
3      return b+1;
4    if (b == 0)
5      return find(a-1, 1);
6    else
7      return find(a-1, find(a, b-1));
8  }
```

a. Show the complete recursive process for the answer **`find(1,5)`**. **[9 marks]**

b. Based on the recursive function above, what is the output of the following statement:
`cout << find (1,5);` **[1 mark]**

c. What is the base case and general case in the above program? **[3 marks]**

d. What is the difference between base case and general case? **[2 marks]**

Question 3 **[15 Marks]**

a. Give the following expression, write the part of the expression that represented by the name of growth rates in the table below. For example, 2 is the constant time in the growth rates.

$2\log_2 n + n\log_2 n + 2n(n-1)/2 + 10n + 2^n + 2$ **[5 marks]**

Name of Growth Rates	**Part of the Expression**
Constant time	*2*
Logarithmic time	
Linear time	
Log-linear time	
Quadratic time	
Exponential time	

b. Calculate the number of steps and find the Big-O notation of the following program segment:
[10 marks]

Statement		**Number of Steps**
1	`for (i = 0; i < n; i++) {`	
2	`  for (j = 1; j <= n; j = j * 2)`	
3	`    cout << "***";`	
4	`      for (j = 0; j < n; j++)`	
5	`        cout << "***";`	
6	`}`	
Total steps		
Big O notation		

Question 4 **[20 Marks]**

a. Given the following array named, **data1**, show the process of sorting the **data1** array into descending order using insertion sort technique. **[10 marks]**

array index	[0]	[1]	[2]	[3]	[4]	[5]
	7	12	24	4	19	32

data1 array

Show your answer by completing the Table below.

Initial Array		After Pass 1		After Pass 2		After Pass 3		After Pass 4		After Pass 5	
[5]	32	[5]		[5]		[5]		[5]		[5]	
[4]	19	[4]		[4]		[4]		[4]		[4]	
[3]	4	[3]		[3]		[3]		[3]		[3]	
[2]	24	[2]		[2]		[2]		[2]		[2]	
[1]	12	[1]		[1]		[1]		[1]		[1]	
[0]	7	[0]		[0]		[0]		[0]		[0]	

b. Given an array **data2 [8, 22, 4, 12, 1]** that will be sorted into ascending order. Use improved bubble sort to sort array **data2** in the table below:

i. Show the results for each pass. **[8 marks]**

ii. Give the total number of comparisons and swaps. **[2 marks]**

array index	[0]	[1]	[2]	[3]	[4]
	8	22	4	12	1

data2 array

Intial Array		**After Pass 1**		**After Pass 2**		**After Pass 3**		**After Pass 4**	
[4]	1	[4]		[4]		[4]		[4]	
[3]	12	[3]		[3]		[3]		[3]	
[2]	4	[2]		[2]		[2]		[2]	
[1]	22	[1]		[1]		[1]		[1]	
[0]	8	[0]		[0]		[0]		[0]	
Total Comparison									
Total Swap									

Question 5 **[20 Marks]**

Given the following **myData** array, answer questions a and b.

array index	[0]	[1]	[2]	[3]	[4]	[5]
	49	71	83	19	45	27

myData array

a. Draw the **merge sort** diagram to sort the **myData** array into ascending order. **[5 marks]**

b. **Program 2-B** is a quick sort function which consists of two functions: **quicksort()** and **partition()**. **[15 marks]**

```
1  //Program 2-B
2   int partition(int T[], int first,int last)
3   {     int pivot, temp;
4         int loop, cutPoint, bottom, top;
5         pivot= T[first];
6         bottom=first; top= last;
7         loop=1;    //always TRUE
8         while (loop) {
9                 while (T[top]>pivot)
10                {  top--;          }
11                while(T[bottom]<pivot)
12                {  bottom++;       }
13                if (bottom<top) {
14                   temp = T[bottom];
15                   T[bottom] = T[top];
16                   T[top] = temp;
17                }
18                else {
19                   loop = 0;
20                   cutPoint = top;
21                }
22        }//end while
23       return cutPoint;
24  }
25
26  void quickSort(dataType arrayT[],int first,int last)
27  {
28       int cut;
29       if (first<last){
20             cut = partition(T, first, last);
21             quickSort(T, first, cut);
22             quickSort (T, cut+1, last);
23       }
24  }
```

Program 2-B – Quick Sort Algorithm

Based on the **myData array** and the quick sort algorithm shown in Program 2-B, trace the quick sort's partition work for `cut = partition(myArray,0,5);` until the **first *cut point*** by using item at index 0 as pivot. In the trace diagram, show the following steps:

i. The movement of bottom and top
ii. The swapping process
iii. The **cutPoint** and the new two parts array for next recursive call.

```
cut = partition(myData,0,5);
```

pivot value = ____	**initial bottom index = ____**	**initial top index = ____**

Array index	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**
Array item	49	71	83	19	45	27

Current value of **bottom** and **top**

bottom = ___ **top** = ___

Array index	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**
Array item						

Current value of **bottom** and **top**

bottom = ___ **top** = ___

Array index	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**
Array item						

Current value of **bottom** and **top**

bottom = ___ **top** = ___

Array index	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**
Array item						

Current value of **bottom** and **top**

bottom = ___ **top** = ___

Array index	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**
Array item						

Current value of **bottom** and **top**

bottom = ___ **top** = ___

Array index	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**
Array item						

Current value of **bottom** and **top**

bottom = ___ **top** = ___ **cutPoint** = ____

Array index	**[0]**	**[1]**	**[2]**	**[3]**	**[4]**	**[5]**
Array item						

Recursive function and array value after **cutPoint** is returned.

```
quickSort(myData, __ , __);
quickSort(myData, __ , __);
```