Data Structures and Algorithms – Pre-midterm exam Review 2021

1. **Given the declaration: char a, *b = &a, **c = &b;**
   Using variable c, write the C statement that will store from keyboard input the value of a.

2. **The following operations can be done using structures but not in arrays: (Select all that apply)**

   A. **Whole data structure assignment using =**
   B. **Pass data structure to a function by copy and by address**
   C. **Return a local data structure to the calling function**

3. **A linked list with head pointer L is pointing to the first of 5 nodes, each node containing an array of 32 characters and a pointer to the next node of the list. At least how many bytes are allocated in the heap memory for list L?**

4. **Given the declaration:**

   ```
   int arr[5][5] = {{-1}, {-2, -3}, {-4, -5, -6}, {-7, -8, -9, -10}};
   ```

   True or false: `arr == *arr`
   True or false: `arr[2] == arr[2][0]`

5. **Given a list in array implementation:**

   ```
   typedef struct{
      int data[0XF];
      int numitems; /*holds number of active elements*/
   }LIST;
   ```

   Fill in the blanks to complete the code of a function which deletes the element in a position corresponding to its index in the given array list.

   ```
   void deleteAtPosition(LIST *A, int P){
   int x;
   if(___(1)___){
       ___(2)___
       for( x = P + 1; ___(3)___; x++){
            ___(4)___
       }
   }
   ```

6. **In which of the following implementations of list are sequential access of data only possible? (Select all that apply)**

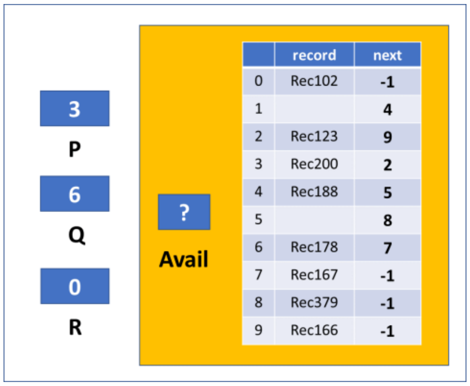
   A. Array implementation   B. Linked list implementation   C. Cursor-based implementation

7. **Which of the following is/are FALSE?**
   1. **Last-in-first-out type of computations are efficiently supported by QUEUES.**
   2. **Implementing QUEUES on a circular array is more efficient than implementing QUEUES on a linear array with two indices. 1**
   3. **First-in-first out types of computations are efficiently supported by STACKS. 0**
   4. **Implementing LISTS on linked lists is more efficient than implementing LISTS on an array for almost all the basic LIST operations.**

8. Refer to the image below. V is a virtual heap with an Avail field containing the index of the next vacant cell and an array with each cell containing a record and an integer indicating the index of the next field. P, Q, and R are head pointers of their corresponding list of characters.

   A) How many available cells are there in the virtual heap (marked in orange)?

   B) What should be the value of the Avail field in the virtual heap structure (marked in orange)?



	record	next
0	Rec102	-1
1		4
2	Rec123	9
3	Rec200	2
4	Rec188	5
5		8
6	Rec178	7
7	Rec167	-1
8	Rec379	-1
9	Rec166	-1

9. On linked list implementation of stack, which of the following statements is/are FALSE?

   A. In push operation, if new nodes are inserted at the beginning of linked list, then in pop operation, nodes must be removed from end.
   B. In push operation, if new nodes are inserted at the end of linked list, then in pop operation, nodes must be removed from the beginning.

10. A single array A[MAX] is used to implement two stacks. The two stacks grow from opposite ends of the array. Variables top1 and top2 (top1< top 2) point to the location (index) of the topmost element in each of the stacks. If the space is to be used efficiently, Identify a condition for "stack full".

11. Consider the code below:

```
void push (int); /* push the argument on the stack S */
int pop (void); /* pop the top of the stack S*/
Stack S;
void main () {
   int c, m, n, r;
   while ((c = getchar ()) != '\n'){
      if (isdigit (c) )
         push (c);
     else if ((c == '+') || (c == '*')) {
         m = pop ();
         n = pop ();
         r = (c == '+') ? n + m : n*m;
         push (r);
     }
   }
  printf("%c", pop ());
}
```

Given the input: 5 2 * 3 3 2 + * +
Determine the output.

12. **Suppose a stack is to be implemented with a linked list instead of an array. What would be the effect on the <u>time complexity</u> of the push and pop operations of the stack implemented using linked list (Assuming stack is implemented efficiently)?**

13. **Suppose a queue is to be implemented with a linked list instead of an array. The rear element is found at the beginning of the list and then the front element is found at the end of the list. What would be the effect on the <u>time complexity</u> of the <u>enqueue and dequeue</u> operations of the queue implemented using linked list?**

14. **The seven elements A, B, C, D, E, F and G are pushed onto a stack in reverse order, i.e., starting from G. The stack is popped five times and each element is inserted into a queue.Two elements are deleted from the queue and pushed back onto the stack. Now, one element is popped from the stack. The popped item is ________.**

15. **Consider the following sequence of operations on an empty stack.**
    push(54); push(52); pop(); push(55); push(62); s=pop();
    Also, consider the following sequence of operations on an empty queue.
    enqueue(21); enqueue(24); dequeue(); enqueue(28); enqueue(32); q=dequeue();

    What is s + q?

16. **Given a queue implemented using circular array with MAX elements, initialized as empty by setting the value of the front field pointing to an index 1 higher than that of the rear field is pointing to, and is considered full when there are MAX - 1 elements. Elements are populated in a clockwise direction.**

```
#define MAX 2021
typedef struct{
   char* data; /*dynamically allocate memory to store MAX elements*/
   int front;
   int rear;
}Queue;
Queue Q;
```

Complete the function definition below which returns 1 if the queue is full and 0 if its not.

```
int isFull(Queue *Q){
    return (__________) ? 0 : 1;
}
```

17. **A queue is implemented using circular array with 8 elements. Suppose that the elements are populated and removed in a clockwise manner, and there are elements populated such that the front element is at index 2 and the rear element is at index 4.**

    A) How many elements are there in the queue?
    B) The following sequence of operation is performed:
    One value is enqueued -> Two values are dequeued -> Three values are enqueued -> One value is dequeued.

    (i) In what index is the front element on now?

    (ii) In what index is the rear element on now?

18. **Digits 1,2,3,4 are entered as an input sequence. Which of the following is a possible output sequence by only using a single stack?**

    A. 1,4,2,3 B. 2,4,1,3 C. 3,2,4,1 D. None of these

19. **In cursor-based implementation of a queue, front and rear pointers are tracked. Which of these pointers is/are guaranteed to be changed while inserting an element into a queue?**

    A. Front pointer B. Rear pointer C. Both front and rear pointer D. Neither front nor rear pointer

20. **Given a pseudo code consisting of stack functions:**

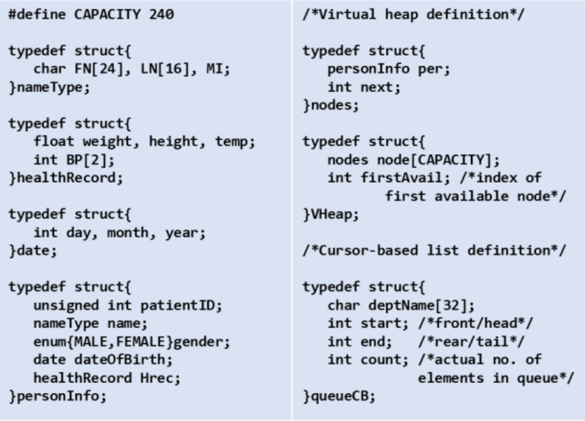
```
void function1(int n)
{
   Stack S;
   initialize(&S);
   while (n > 0)
   {
      push(&S, n%2);
      n = n/2;
   }
   while (!isEmpty(S))
      printf("%d", pop(&S));
}
```

What does the above function do in general?

- **A. Prints binary representation of n in reverse order**
- **B. Prints binary representation of n**
- **C. Prints the value of Log_2(n)**
- **D. None of these.**

Programming questions:

21. **CFP Hospital consists of 7 departments, and each department contains walk-in patients waiting for their corresponding consultations or treatments to be done. The manager decided to create a program which assigns each department with a queue that help identify the sequence of people to be served next. Furthermore, he opted for a cursor-based implementation with the number of nodes initialized in the virtual heap representing the hospital's overall capacity (as part of the COVID-19 protocol) which will be utilized by all the queues. See definition below.**



```
#define CAPACITY 240

typedef struct{
   char FN[24], LN[16], MI;
}nameType;

typedef struct{
   float weight, height, temp;
   int BP[2];
}healthRecord;

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

typedef struct{
   unsigned int patientID;
   nameType name;
   enum{MALE,FEMALE}gender;
   date dateOfBirth;
   healthRecord Hrec;
}personInfo;
```

```
/*Virtual heap definition*/

typedef struct{
   personInfo per;
   int next;
}nodes;

typedef struct{
   nodes node[CAPACITY];
   int firstAvail; /*index of
          first available node*/
}VHeap;

/*Cursor-based list definition*/

typedef struct{
   char deptName[32];
   int start; /*front/head*/
   int end;   /*rear/tail*/
   int count; /*actual no. of
              elements in queue*/
}queueCB;
```

Write the code of functions:

A) **enqueue(). The function inserts a given patient information into the rear of the given queue.**

B) **removeFirst(). The function will remove the first element of a given queue and return the data contained in it to the calling function. A dummy variable containing "-", '-', and 0 for string, character, and numeric fields respectively will be returned instead if the given queue has no first element.**

Note: The function cannot call any existing function.

22. GSM Academy offers 2 online courses for interested students to enroll in: CISCO1 and CISCO2. The academy director already has a generic list of all the students registered this year stored using linked list implementation. A teacher, sir G handling the two courses is requesting for a combo list which will contain the lists of CISCO1 and CISCO2 students who are qualified based on their scores in the admission exam. (Passing score is 50).

    The lists to be created for CISCO1 and CISCO2 will be stored using array and linked list implementations respectively, and both of them are sorted in order according to ID as well.

```
#define SIZE 512

typedef struct{
   char FN[24], LN[16], MI;
}nameType;

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

typedef struct{
   char studID[8];
   nameType name;
   enum{MALE,FEMALE}gender;
   enum{CISCO1,CISCO2}course;
   date enrolled;
   float admissionExamScore;
}personInfo;
```

```
typedef struct cell{
   personInfo studs[SIZE];
   int cnt; /*actual number of
             students*/
}*Array;

typedef struct node{
   personinfo stud;
   struct node* link;
}*LinkedList;

/*Combo list definition*/
typedef struct ComboList{
   Array CISCO1;
   LinkedList CISCO2;
   int numQualifiedStudents; /*for
           both courses combined*/
}CL;
```

    Write the code of function createComboList() which removes the qualified students from the given unsorted generic list and have their records transferred to their proper locations in a newly created combo list as indicated above. The combo list shall then be returned to the calling function.