Data Structures and Applications (CSE-2152) – I Test 2019 Scheme

MCQ

Q1. What is the output of the following program segment assuming that the machine address takes 4 bytes? (1/2)

```
int main()
{
    char *s1 = "manipal";
    char s2[] = "manipal";
    int *p, a;
    p = &a;
    printf("%d %d %d", sizeof(s1), sizeof(s2), sizeof(p));
}
```

1. 4 8 4
2. 8 8 2
3. 1 8 4
4. 2 7 4

Q2. A user defined data type, which is used to assign names to integral constants is called: (1/2)

1. typedef
2. array
3. structure
4. enum

Q3. **What is the output of the following program?**

```
#include<stdio.h>
int rec_call(int a, int b)
{
 If (b==1)
        return a;
 else
        return a + rec_call(a,b-1);
}
int main()
{
printf("%d",rec_call(2,4));
}
```

1. 6
2. 12
3. 8
4. 16

Q4. Given the declaration, struct part{int a; int b;} *my_part; which of the following is an incorrect way of referring structure member? (1/2)

1. *my_part.a = 10;
2. (*my_part).a = 10;
3. my_part->a = 10;
4. Both 2 and 3

Q5. The equivalent pointer expression for a[i][j] is (1/2)

1. (**(a+i)+j)
2. *(*(a+i)+j)
3. (*(*a+i)+j)
4. *(a+*(i+j))

Q6. Which data structure is used by the system for invoking a recursion call?(1/2)

1 Queue
2 List
3 Stack
4 Array

Q7. In an integer sequential queue of MAX_SIZE = 5 and with initial value of -1 in rear and front, what is the value of front and rear after the following sequence of operations: (1/2) addq(1); addq(2); addq(3); deleteq(); addq(5); deleteq(); deleteq():

1 -1, 2
2 2, 3
3 4, 3
4 4, -1

Q8. Which of the following is not the natural application of Stack? (1/2)

1 Palindrome Test
2 Decimal to Binary conversion
3 Evaluation of postfix expression
4 User applications waiting for execution by CPU

Q9. The prefix form of the expression a*(b+c)/d is (1/2)

1. /*a+bcd
2. /*+abcd
3. /a*+bcd
4. /a*b+cd

Q10. The postfix form of the expression (c*(d-e) +f) *g is (1/2)

1. cde*f-+g*
2. cde*-fg+*
3. cdefg-*+*
4. cde-*f+g*

Q11.	Create a type STUDENT which is used to represent a student structure with	(2)

	reg_no, name and cgpa. Write a complete program to dynamically allocate memory for N such students, where N is read from keyboard and to read and display information for each student.
	
	**#include<stdio.h>
	#include<stdlib.h>
	typedef struct
	{
	int reg_no;
	char name[20];
	float cgpa;
	}STUDENT;** --------
	**int main()
	{
	int i, n;
	printf("Enter the number of students\n");
	scanf("%d", &n);
	STUDENT *s=(STUDENT *)calloc(n, sizeof(STUDENT));** --------
	**printf("enter the students information\n");
	for(i=0; i<n; i++)
	scanf("%d%f%s", &(s+i)->reg_no, &(s+i)->cgpa, (s+i)->name);** --------
	**for(i=0; i<n; i++)
	printf("%d\t%f\t%s\t/n", (s+i)->reg_no, (s+i)->cgpa, (s+i)->name);** ---------
	}
		½ M
		
		½M
		
		½M
		
		½ M

Q12	Write an algorithm to convert a prefix expression to postfix. Show the working of the algorithm on the string *+abc. **Algorithm for Prefix to Postfix:** ▪ **Read the Prefix expression in reverse order (from right to left)** ▪ **If the symbol is an operand, then push it onto the Stack** ▪ **If the symbol is an operator, then pop two operands from the Stack Create a string by concatenating the two operands and the operator after them. string = operand1 + operand2 + operator And push the resultant string back to Stack** ▪ **Repeat the above steps until end of Prefix expression.** ------- Prefix string: *+abc Postfix equivalent: ab+c* --------	(2) 1M 1M
Q13	Write a complete program to find the solution for tower of Hanoi problem using recursion.	(3)

Token	Stack
c	C
b	c b
a	c b a
+	c ab+
*	ab+c*

```
#include <stdio.h>
void towers(int, char, char, char);
int main()
{
   int num;
   printf("Enter the number of disks : ");
   scanf("%d", &num);                                          --------   ½M
   printf("The sequence of moves involved in the Tower of Hanoi are :\n");
   towers(num, 'S', 'A', 'D');                                 --------   ½ M
   return 0;
}
void towers(int num, char S, char A, char D)
{
   if (num == 1)
   {
      printf("\n Move disk 1 from peg %c to peg %c", S, D);    --------   ½ M
      return;
   }

  towers(num - 1, S, D, A);                                    ----------  ½ M

    printf("\n Move disk %d from peg %c to peg %c", num, S,D);  ----------  ½ M

   towers(num - 1,A,S,D);                                      ----------  ½ M
}
```

Q14 Write a complete program to implement a stack of characters. Using this stack, check if the given expression has properly matching opening and closing parenthesis. (3)

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

#define MAX_STACK_SIZE 20
char Stack[MAX_STACK_SIZE];
int top=-1;

bool IsEmpty(void){
   return(top < 0);
}

void IsFull(void) {
   if(top >= MAX_STACK_SIZE-1) {
      printf("Stack is Full\n");
      exit(-1);
   }
}                                                              ----   ½ M

void push(char c){
   IsFull();
   Stack[++top]=c;
}
```

```
char pop(void){
   if(IsEmpty()) {
      return '#';
   }
   return Stack[top--];
}
```

------ 1M

```
int main() {
   int i, n;
   char exp[20], token_e, token_s;
   int balanced=1;
   printf("Enter the expression: ");
   scanf("%s", exp);
   n=strlen(exp);
   for(i=0; i<n; i++) {
      token_e=exp[i];
      if(token_e=='(') push(token_e);
      else {
         if(token_e==')') token_s = pop();
         if(token_s=='#') balanced=0;
      }
   }
   if(IsEmpty() && balanced==1)
      printf("\n Expression has balanced paranthesis");
   else
      printf("\n Expression has imbalanced paranthesis");
   return(0);
}
```

----- ½ M

------ 1M