Shahjalal University of Science and Technology
Department of Computer Science and Engineering
1^{st} year 1^{st} Semester Final Examination—June 2019 (Session 2018-19)
Course No.—**CSE 133**
Course Title—**Structured Programming Language**
Time—**3 Hours** Credit: **3.00** Total Marks#**100**

(Answer All the Questions)

Group A

1. Answer the following Questions in short (Any **Five**). $5 \times 2 = 10$

   (a) Add just one line in the following program to compile it and print:

   ```
   It is really an awesome match. - said by the teacher
   Where the result is unknown! - said by the teacher
   ```

   The program:

   ```
   #include <stdio.h>

   int main()
   {
                                       disp(It is really an awesome match.);
                                       disp(Where the result is unknown!);
                                       return 0;
                                   }
   ```

   (b) `[^cse]`—Where is it used? What is the purpose of this using? What is the other use of **circumflex**?

   (c) Would the following C code block generate any compilation error? If the answer is yes, then write the reason with corrected code block. [Assume that, corresponding header files are included and main function is written properly.]

   ```
   typedef struct node{
       int val;
       node *next;
   } node;
   node *start;
   ```

   (d) An array is declared by `long long int x[3][1][10][34][21]`. It is given that the memory address of the element `x[1][0][7][5][11]` of this array is **C5D0208**. What would be the memory address of the last element of this array?

   (e) How could `#ifdef` help you to do some extra work (like printing something or taking input from file etc.) in your PC rather than other PC while executing exactly the same program?

   (f) What is the difference between **Compilation Error** and **Runtime Error**? What would be the output for `char s[]="53.26"; printf("%05.4s\n", s);` ?

   (g) Which logical expression is associated with the else clause for **if(e1) if(e2) s1 else s2**?

   (h) What are the three conditions a **Recursion** must fulfill? What is the difference between **7238ULL** and **7238L**?

2. Answer the following Questions (Any **Four**). $4 \times 5 = 20$

   (a) Suppose, you are given a binary floating point number 1101101110001110.101110001; you are also given 2 bytes to store it where 8 bits for mantissa and 7 bits for exponent. Sketch the memory representation. Reconstruct the number retrieving it from this memory representation. Is there any data loss? If yes, then what is the order of this loss? You can write the result in any base of your choice. $2 + 2 + 1 = 5$

   (b) Write a complete C program to determine the value of the nth Fibonacci number, F_n where $F_n = F_n - 1 + F_n - 2$ and $F_1 = F_2 = 1$.
   Let the value of n be an input quantity. 5

   (c) Write a program to find out the **GCD** of two integers **a** and **b** using **recursion**. 5

   (d) For input **12324322342** what would be the value of a,b,c,d after executing the code—`int a=1,b,c,d=9;scanf("%2d%*3d%1d%4d",&a,&b,&c,&d);`? Suppose, A ball fits in a cubic box smoothly and it just touches edges of the box at all it's 6-side. Now, you are given a positive real number as input which is the height of the cubic box. Now, write a program to find the volume of empty spaces in the box and report it in the output. 2+3=5

   (e) Observe the following two structure declarations for a **Doubly Linked List**. Are they syntactically correct? Compare them visualizing their memory representations.

i)
```
struct node{
      int val;
      struct node *next, prev;
  };
```

ii)
```
struct node{
      int val;
      struct node *next, *prev;
  };
```

(f) What is the difference between declaring `k` as `union {char a[10]; double b;}k;` and `struct {char a[10]; double b;}k;`?
`int x=___;printf("%d\n",x);`—What would be happened if the gap is filled by the following value? Explain each scenario. 2+3=5

i. 54,432 ii. 1981181 iii. 0127 iv. 0169 v. 3.1416 vi. 0X6F

3. Answer the following Questions (Any **Two**). $2 \times 10 = 20$

(a) Write a code to print all the prime divisors of a given number (from standard input). Your code should work fine for numbers ranging 10^{12}. [Hint: Generate all primes within 10^6 using **sieve of eranthoses**. Then use these primes to factorize.]

(b) Write the output of the following program:

```
#include <stdio.h>

int bar(int x)
{
  if(x&2&&x>0)printf("%05d ", x);
  if(x<0||x>10) return -x;
  int ret = 0;
  switch(x)
  {
    default:
      ret+=bar((x-->4)-3)+bar(x);
    case 4:
      ret-=bar(x+13);
      break;
    case 0:
      ret%=bar(x-2);
  }
  return ret;
}
```

```
int main()
{
  int y = 0,n=8;
  puts("Let make sth different!");
  printf("Recursion means %#4o!\n",n);
  if( ((1<<4)&n) == 1 ) puts("No!");
  for (n=~n;;y++)
  {
    puts("Calling...");
    if(y%3) continue;
    printf("bar(%d)=%d\n",y,n=bar(y));
    if(y>n) break;
    printf("Milestone#%03d\n",y);
  }
  printf("The World is too tired!\n");
  return 0;
}
```

(c) Implement a Queue using Pointer (Linked List). Write three functions `push`, `pop` and `print_array`. `push` and `pop` function must have constant complexity and `print_array` must have linear complexity. [Hint: Take two global pointer of structure `front` and `rear`.]

Group B

4. Answer the following Questions in short (Any **Five**). $5 \times 2 = 10$

(a) If you want to take input from one file and want to give output to another file instead of standard I/O then what are the two lines you have to write? Remember, all other lines would remain unchanged.

(b) What are the two data types which could be used in **bit field** in all C compiler? Define a structure named **date** where **month**, **day** and **year** would take 2, 2 and 4 bits respectively.

(c) What is the difference between **(y!=2019)** and **(y=!2019)**? Write an expression to generate the mask **FFFE0**. You can use only bitwise operators and minus(-).

(d) Suppose, an integer variable is accessing like this `cse.iict->eee->swe.cse2018` Now, identify the pointer variables and normal variables.

(e) `int kk = sqrt(48.99);printf("%#+-015.0e:\n",(float)kk);`—What would be printed by this piece of code? [If a portion depends on OS and/or some other parameters, then assume that it would take 3 decimal places.]

(f) `int x[10]={12,53},*p=x+1;*p++;printf("%+05X\n", 10+++*p);`—Would it be any compilation error? If yes, then add just one letter to resolve it. What would be the output then?

(g) What is **Generality**? `int p=32,q=30;p^=q^=p^=q;printf("%0,%o",p,q);`— What would be the output?

(h) What is **Simplicity**? Let (B)`const static cse18=19;` and (A)`static const int cse18=19;` —Which one is syntactically correct?

i) A *ii*) B *iii*) both of them *iv*) none of them

5. Answer the following Questions (Any **Four**). $4 \times 5 = 20$

(a) Write a program that takes an integer number **n** from standard input which ranges from 1 to 10^{18} and gives another integer **m** in output. $m = n - 10^x$ where **x** is the number of 0's in n. Example: for input **1203001**, the output would be **1202001**.

(b) Find the output of the following program showing the calculations.

```
#include<stdio.h>
int main()
{
    int x[5]={2,1,5,3}, y, *p, *q, *r;
    p = x;q = &y;
    *q=*(x+4);
    r = ++p;
    (*p)++;q++;r--;
    printf("%d,%d,%d,%d,%d\n",*x,*p,*q,*r,*(x+4));
    *q = (*p<<3)-1;
    y = *p+*q+*r+*(r+4);
    r+=4;q=p; *r=*q+13-*(p-1);
    if(q==x+1)  p = &y;
    printf("%d,%d,%d,%d,%d\n",*x,*p,*q,*r,*(x+1));;
    return 0;
}
```

(c) Is there any difference between the declarations `char cse[]={'1','8'};` and `char cse[]="18";`? Show the reason behind your answer. Write the output of the following program. 1+4=5

```
#include <stdio.h>
#define CSE 4+6
#define SUST 9-CSE+CSE
int main()
{
    int p = 1;
    int N = (~(~((12*SUST*2)<<2)-p))%420;
    printf("%010X\n",++N+10);
    return 0;
}
```

(d) Write C program using recursive function to display a given string reversely. 5

(e) What is the complexity of **Binary Search**? Write the body of this function: `int search_names(char names[][100],int n, char name[100]);` Here, *names* is an array of length **n** of sorted names in lexicographical order. You have to find the *name* using Binary Search. Return 1 if it exists, otherwise 0. 1+4=5

(f) Generate the following "pyramid" of digits, using nested loops.[Do not simply write out 10 multi-digit strings. Instead, develop a formula to generate the appropriate output for each line.] 5

```
         1
        232
       34543
      4567654
     567898765
    67890109876
   7890123210987
```

6. Answer the following Questions (Any **Two**). $2 \times 10 = 20$

(a) Write the output of the following program:

```
#include<stdio.h>

int rec(int *u,int v[])
{
    static int y = 90;
    u=&y;v[1]++;++y;
    (*u)++;v[5]=11;
    return *u;
}

int main()
{
    int x = 8,i,y[10]={3,9},z;
    z=rec(&x,y)+rec(&x,y);
    printf("%d %d\n",x,z);
    for (i=y[1]-1; i>=0; i--)
        printf("y[%d]=%d\n",i,y[i]);
    return 0;
}
```

(b) Suppose there are four folders named "Branch A", "Branch B","Branch C" and "Programs" in drive "D:". So, there are three branches identified by A, B and C. Each branch has four sections. So, each of the branch folder has four sub-folders named "Section 1", "Section 2", "Section 3" and "Section 4". In each of the section folder, there are two files named "Revenue.in" and "Expense.out". Now, your task is to write a program file named "Calculate Total Profit.cpp" under the folder named "Programs". Your program would generate a file named "Summary.txt" under the drive "D:" where you would show the sum of revenues and expenses of each branch in separate lines. Then calculate the sum of revenues minus sum of expenses. If the values is non-negative then write a line like "Total Profit = < *value* >", otherwise write "Total Loss = < *value* >". Replace the '< *value* >' portion with the absolute amount of that value. Read files with absolute address and write file with relative address.

(c) Write a C function named **find_median** which takes a list of variable number of integer arguments and returns the middle value of the list after sorting. If the list has even number of elements, then it returns the average of two middle values. The very first argument is an integer which indicates the number of elements in the list. Example: Calling `find_median(5,3,4,5,6,7)` returns **5** and calling `find_median(4,100,3,10,1)` returns **6.5**. Remember, your program must be efficient. [Hint: Write a function which takes variable number of arguments. Take them in an array. Sort them using qsort. Write a comparator function for qsort. If the number of elements is odd, then return the middle value of the array, otherwise return average of the two middle values.]