CS101 Final Examination

Name:

Roll Number:

Write down your name, roll number on the top of the exam sheet in space provided above. Read the question paper carefully and ascertain that you understand every question clearly. Answer the questions in the space provided in the question paper. There is no negative marking.

Q1 / 5	***Q2 / 5***	***Q3 /5***	***Q4 /8***	***Q5/10***	*Q6/12	***Q7/10***	***Q8/5***	***Q9/5***	***Q10/5***
Q11/6	***Q12/10***	*Q13/20	***Q14/5***	***Q15/10***	***Q16/10***	***Q17/5***	***Q18/5***	***Q19/5***	***Q20/5***
Q21/5	***Q22/5***	***Q23/5***	***Q24/10***	***Q25/10***	***Q26/***4	***Q27***	***Q28***	***Q29***	***Q30***

*HARD

Q1. Consider the following program:

```
 int main()
 {
   int* p;
   p = (int*)malloc(sizeof(int));
   *p = 10;
      printf("The value of p is %d\n",%p);

      return 0;
   }
```

Let X = "The C program will work correctly on all machines"
Y = "The C program will work correctly on any machine which can support 1024 GB hard disk."
Z = "The C program will work correctly on any machine which can support up to 512 MB hard disk."
Choose the most *appropriate* option [5 points]:

(1) X, Y, Z (2) X (3) Y (4) Z

(5) Y, Z (6) X, Z (7) X, Y

Q2. Consider the following program:

```
int permanent(unsigned int x)
{
   int permit=0;
```

```
    while(x)
    {
        permit++;
        x = x&(x-1);
    }
    return permit;
}
```

Recall that "a & b" is the value obtained by taking bitwise "and" of a, b. The above function is called with value of x = 55555. What value does it return? [5 points]

Q3. What is the output of the following program ? [5 points]

```
#include <stdio.h>
int main()
{
    long a = 125;
    printf("%d\n", a);
    printf("%d\n", *(int *)&a);

    return 0;
}
```

Q4. Describe the following functions in one sentence each ? [8 points]

fseek(f,1,i):

fread(s,i1,i2,f):

fwrite(s,i1,i2,f):

ftell(f):

Q5 The following routine is supposed to create and return a two dimensional array of integers of size n. Fill in the blank spaces to complete it (Leave empty where you feel you don't need to write anything. [10 points]

```
_____ Create(int n)
{
    int **A, i;

    A = ____________________________;

    ________________________________;

    ________________________________;

    ________________________________;

    ________________________________;

    return A;
}
```

Q6 Let A[100][100] be a two dimensional array which is filled either with ones or zeroes. The following recursive routine determines for given indexes i and j, whether there is a path of Zeroes from i to j. That is, it returns one if there exists indexes a 1, a 2, a 3, ... , a l, for some l such that A[i,a 1] = A[a 1,a 2] = A[a 2,a 3] = ... = A[a l,j] = 0 and zero otherwise. Fill in the blanks. [5+5+2 = 12 points]

```
int findPath(int A[100][100], int i, int j, int count)
{
    if (count <= 0) return 0;

    if (A[i,j] == 0)
        return 1;
    for (int k=0;k < n; l++)
    {
      if(______________________________&& ______________________ && (A[k][j] ==0))
           return 1;
      else continue;
    }
    return ____________;
}
```

Q7. Define a *struct* data type with the following fields: (a) Name (String of potentially arbitrary length) (b) Age (Non-zero integers) (c) lPtr (Pointer to the same record) (d) rPtr (Pointer to the same record) (e)arr (integer array with 10 locations) [2*5 = 10 points]

Q8. Assume that it's a 32 bit bus machine and fields of the record are stored contiguously without leaving any padding space between two fields. How many bytes of space does such a record occupy in the memory? [5 points]

Q9. The following is a simple c program, which doesn't work. Fix it by crossing out ONLY appropriate lines and writing besides them, the new code. [5 points]

```
  #include <stdlib.h>
  #include <stdio.h>
  void Error(char* s)
  {
      printf(s);
      return;
  }

  int main()
  {
      int *p;
      p = malloc(sizeof(int));
```

```
        if(p == NULL)
        {
            Error("Could not allocate the memory\n");
            Error("Quitting....\n");
            return -1;
        }
        else
        {
            /*some stuff to use p*/
        }
        return 0;
    }
```

Q10. What is a potential problem with the following C program? [5 points]

```
    #include <stdio.h>
    int main()
    {
        char str[80];
        printf("Enter the string:  ");
        scanf("%s",str);
        printf("You entered:%s\n",str);

        return 0;
    }
```

Q11. Consider the following function [6 points]

```
#include <stdio.h>
int compute(int u,int v)
{
    int t;
    while(v > 0)
    {
        if(u > v)
        {
            t = u;
            u = v;
            v = t;
        }
        v = v-u;
    }
    return u;
}
```

What does (a) compute(6,64), (b) compute(1232,91) and (c) compute(2^{10000},2^{10}-1) return?

(a) (b) (c)

Q12. Assume you are given implementation of following routines for STACK (LIFO):

(1) void push(char x); The function push() pushes character on the top of a stack.

(2) char pop(); The function pop() pop's topmost character from top of stack.

Complete the following routine which returns 1 iff a string **s**, of length **len** is a palindrome, using stack. That is, string s and its reverse s^R are same. For example, let s = "abracadabra", then its reverse is "arbaadacarba", they are not same. However, "dollod" and "carrac" are palindromes. [10 points]

```
int isPalindrome(char str[], int len)
{
  for (int i=0; i<= len; i++)
  {
     p________;
  }
  for (i = 0; i <= len; i++)
  {
     if (p_______!= str[i])
       return 0;
  }
  return 1;
}
```

Q13. The setting for this question is same as that of Q6. There is an additional property of matrix A[][]: For all i,j: A[i][j] = A[j][i] (transpose of matrix is same as matrix, represents an undirected graph). The following routine isThereCycle() determines if there is a 'cycle' of Zeroes in matrix A. That is, it returns <u>one</u> if there exists indexes <u>a 1</u>, <u>a 2</u>, <u>a 3</u>,.., <u>a l</u>, for some l such that A[<u>i</u>,<u>a 1</u>]=A[<u>a 1</u>,<u>a 2</u>]=A[<u>a 2</u>,<u>a 3</u>]= ....=A[<u>a l</u>,<u>i</u>]=0 and zero otherwise (A path starts at index i and finishes at i). [4+4+3+3+3+3 = 20 points]

Assume the following routines are given to you for manipulating a queue data structure: (a) void Enqueue(int x): Inserts an element "x" in the queue (b) int Dequeue(): Deletes an element from queue and returns it (c) int FrontQueue():Returns the element at front of queue, without deleting it. (d) int IsEmpty(): Returns 1 if queue is empty, and 0 otherwise; The algorithm is: starts with some vertex and keeps entering every new vertex it is connected with to queue and marks it. If a marked vertex is visited again implies cycle. Fill the blanks spaces below and complete the program.

```
==============================================================
int visited[100];
void Initialize(int x[100]); /* Initializes array x[] to all
'0' values */
int areAllVisited(int x[100]); /* checks if array x[] is
marked with '1' values:     returns -1 if all entries are
marked '1' and any index i if 'i' is an unmarked index between
0,99 */

int IsThereCycle(int A[100][100])
{
   Initialize(visited);
   int i=0, k, currentElement, counter;
   _______________; visited[0]=1;

   while (IsEmpty()!= 1)
   {
      currentElement = ____________________;

      for (counter=0;counter<100;counter++)
```

```
        {
            if (counter == currentElement)
                continue;
            if (A[currentElement][counter] == '0')
            {
                if (visited[counter] == 1)
                    return 1;
                else
                {
                    ____________________________;
                    ____________________________;
                }
            }
        }
        if ((k = areAllVisited(visited)) != -1)
        {
            ____________________________;
            ____________________________;
        }
    }
    return 0;
}
```

Q14. With every use of a memory allocation function, what function should be used to release allocated memory which is no longer needed? [5 points]
a. unalloc() b. dropmem() c. dealloc() d. release()
e. free()

Q15 Fill the table below. [2*5 = 10 points]

char *p[2][3] = { "abracadab", "creatin", "lopside", "junkee", "rubbish", "doodling" };

Expression	An equivalent Expression	Value
**p[1]		
**(p[1] + 2)		
***p		
(*(*(p+1) +1))[7]		
*(p[1][2] + 2)		

Q16 Write a routine that takes an array A[] of integers and two pairs of indexes: (s_1,f_1) and (s_2,f_2), where s_1 marks the starting index and f_1 the ending index of a segment of array A[]. s_2 and f_2 are defined in same way. Assume that segments A[s_1,..,f_1] and A[s_2,..,f_2] are sorted in increasing order. Write a routine that merges A[s_1,..,f_1] and A[s_2,..,f_2] and writes into array B[], so that B[] is also sorted in increasing order. [10 points]

```
void merge(int A[], int s_1, int f_1, int s_2, int f_2,int B[])
{









}
```

Q17 What is the error in this code? [5 points]

```
#include <stdio.h>
int main(void)
{
    char Alpha[]="ABCDEFGHIJKLMNOPQRSTUVWXYZ";
    char alpha[27]="abcdefghijklmn";

    alpha = Alpha;
    printf( "alpha is now %s\n", alpha);

    return 0;
}
```

(a) Array incorrectly initialized

(b) illegal assignment

(c) Variable names spelled the same

(d) printf needs an address of operator

(e) Insufficient memory

(f) Segmentation fault.

Q 18. Consider the following code. [5 points]

```
void *ptr;
myStruct myArray[10];
ptr = myArray;
```

Circle the correct way to increment the variable "ptr"?

Choice 1 ptr = ptr + sizeof(myStruct);

Choice 2 ++(int*)ptr;

Choice 3 ptr = ptr + sizeof(myArray);

Choice 4 increment(ptr);
Choice 5 ptr = ptr + sizeof(ptr);

Q19 Consider the following code. [5 points]

```
char ptr1[] = "Hello World";
char *ptr2 = malloc( 5 );
ptr2 = ptr1;
```

What is wrong with the above code (assuming the call to malloc does not fail)?
Choice 1 There will be a memory overwrite.
Choice 2 There will be a memory leak.
Choice 3 There will be a segmentation fault.
Choice 4 Not enough space is allocated by the malloc.
Choice 5 Program will not compile, saying illegal assignment.
Choice 6 Insufficient memory.

Q 20.What does the "auto" specifier do? [5 points]
Choice 1: It automatically initializes a variable to 0;.
Choice 2: It indicates that a variable's memory will automatically be preserved.
Choice 3: It automatically increments the variable when used.
Choice 4: It automatically initializes a variable to NULL.
Choice 5: It indicates that a variable's memory space is allocated upon entry into the block.

Q 21.How is a variable accessed from another file? [5 points]
Choice 1: The global variable is referenced via the extern specifier.
Choice 2: The global variable is referenced via the auto specifier.
Choice 3: The global variable is referenced via the global specifier.
Choice 4: The global variable is referenced via the pointer specifier.
Choice 5: The global variable is referenced via the ext specifier.

Q 22.

```
int x[] = {1, 2, 3, 4, 5};
int u;
int *ptr = x;
????
for( u = 0; u < 5; u++ )
{
   printf("%d-", x[u]);
}
printf( "\n" );
```

Which one of the following statements could replace the ???? in the code above to cause the string "1-2-3-10-5-" to be printed when the code is executed? [5 points]
Choice 1 *ptr + 3 = 10;
Choice 2 *ptr[3] = 10;
Choice 3 *(ptr + 3) = 10;
Choice 4 (*ptr)[3] = 10;
Choice 5 *(ptr[3]) = 10;

Q23 Consider the program below. [5 points]

```
#include <stdio.h>
void func()
```

```
{
    int x = 0;
    static int y = 0;
    x++; y++;
    printf( "%d -- %d\n", x, y );
}
int main()
{
    func();
    func();

    return 0;
}
```

What will the code above print when it is executed?

Choice 1	1 – 1	1 -- 1
Choice 2	1 – 1	2 -- 1
Choice 3	1 – 1	2 -- 2
Choice 4	1 – 0	1 -- 0
Choice 5	1 – 1	1 – 2

Q 24.Write a function that takes as input two matrices A and B of dimensions n * n with integer entries and computes their product and stores in matrix C. Assume that memory allocation and deallocation is done outside the function. [10 points]
Hint: (1) A[i][j] = *(*(A+i) j); (2) How is C[i][j] defined in terms of entries of A and B?

```
void matMult(int **A, int **B, int **C, int n)
{

}
```

Q25. A Binary Search Tree(BST) is a structure for storing data such that interactive search, insertion and deletion can be performed efficiently. Let us define a node in a BST as a structure in the following way:

```
struct bstnode{
    int key;
    struct bstnode *leftp;
    struct bstnode *rightp;
}
```

The left pointer points to the left subtree of the node and the right pointer points to the right subtree. Any node in a BST has the property that any element stored in the left subtree of a node is less than the value stored in the "key" and any element in the right subtree of the node is greater than this value.

Write recursive functions for search() ~~and print()~~ in this data structure. search() searches for a key in a BST and returns 1 if the key is found, else returns 0. ~~print() prints all the elements of the binary search tree in sorted order~~. [10+~~10=20~~ points]

```
int search(int key, struct bstnode *bst)
{





}
```

```
~~void print(struct bstnode *bst)~~
{




}
```

Q26 Put these stages of software development in order:- [4 points]

a. Link
b. Compilation
c. Execute
d. Precompilation
e. Indentation and Documentation