



33.5

**INDIAN INSTITUTE OF TECHNOLOGY ROPAR**  
**Department of Computer Science & Engineering**  
**GE103 Introduction to Computing & Data Structures**  
**MidTerm Exam 05 Oct 2018**

Max. Marks: 40

Time Limit: 120 minutes

Name: \_\_\_\_\_

Roll No: \_\_\_\_\_

**NOTE:**

- Read the questions carefully, and write your answers as neatly as possible.
- You need to write your answers in the space provided below each question. No extra sheet should be attached to this paper. Rough work may be done in the space provided or in last empty sheet.
- Best wishes !

1. [2 marks] Consider a two dimensional array : A[6][8] of total 48 integer elements. If the base address (A) is 1600 and the system uses zero-indexing, what is the memory address of element A[3][4] ? Assume:

(a) Row-major order

Assuming size of integer to be 4 bytes

$$\begin{aligned} \text{address} &= 1600 + (3 \times 8 + 5) \times 4 \\ &= 1600 + 116 \\ &= 1716 \end{aligned}$$

(b) Column-major order

$$\begin{aligned} \text{address} &= 1600 + (4 \times 6 + 4) \times 4 \\ &= 1600 + 112 \\ &= 1712 \end{aligned}$$

2. [18 marks] What will be the output for the following codes. Explanation for the output not necessary.

**Answers / OUTPUT**

```
#include <stdio.h> /* 2 marks */
int main() {
    float f1; int i=40, j=30, k=20;
    int p=5;
    f1=42/4 + 4.0/3 + 5.24;
    p = i>j>k;
    printf("f1= %.2f p=%d", f1,p);
}/* You may use the space here for rough work/calculations */
```

$$\begin{aligned} 10 + 1.33 + 5.24 &= 16.57 \\ p &= 0 \end{aligned}$$

f1=16.57 p=0

✓  
2

## Answers / OUTPUT

```
#include<stdio.h> /* 2.5 marks */
```

```
void main() {
    char arr[] = {'l', 'a', 't', 'e', 's', 't'}; //First element is L lower case
    char *p = (arr+2);
    printf("%c", *p+2);
    printf("\n %d %d", sizeof(arr), sizeof(p));
}/* You may use the space here for rough work/calculations */
```

7

✓

6 4

2.5

```
#include <stdio.h> /* 2.5 marks */
```

```
void main()
{
    for (int k=1; k<4; )
        printf( "%d \n", ++k );
}/* You may use the space here for rough work/calculations */
```

2

3

4

✓

2.5

```
# include <stdio.h> /* 3 marks */
```

```
int main() {
    int i = 0;
    for (i=1; i<20; i++) {
        switch(i) {
            case 1:
                i += 1;
            case 2:
                i += 3;
            case 4:
                i += 4;
            default:
                i += 8;
                break;
        }
        printf(" %d ", i);
    }
    return 0;
}/* You may use the space here for rough work/calculations */
```

17 18 26

17

18

26

2

4

~~1726~~

1726

space here

3

```
#include <stdio.h> /* 2 marks */
```

```
#define ALPHA 0
```

```
#define BETA 1
```

```
int main() {
```

```
    int i = 5;
```

```
    switch (i & 1)
```

```
    {
```

```
        default: printf("Default");
```

```
        case ALPHA: printf("alpha"); break;
```

```
        case BETA: printf("beta"); break;
```

```
    }
```

```
    return 0;
```

```
    /* You may use the space here for rough work/calculations */
```

101  
0001

Answers / OUTPUT

beta

2

```
#include <stdio.h> /* 3 marks */
```

```
int main(){
```

```
    int k, sum=0;
```

```
    for (k=2048; k; k >= 1)
```

```
        sum++;
```

```
    printf("%d %o %x ", sum, sum+1, sum+2);
```

```
    return 0;
```

```
    /* You may use the space here for rough work/calculations */
```

12

14

2 2048  
2 1024  
2 512  
2 256  
128  
111111110  
011111111  
8 13 5  
8 1 0

2 10  
11  
01  
00

12 15

E

3

```
#include <stdio.h> /* 3 marks */
```

```
void main()
```

```
{ int i=1, j=5, k=11;
```

```
    int *p = &j; int *q = p; int *r = &k;
```

```
    *p = i; (*p)++;
```

```
    i += 2;
```

```
    *r = *r - *q;
```

```
    p = r; j = j + i;
```

```
    k = k + *q;
```

```
    printf("%d %d %d", i, j, k);
```

```
    /* You may use the space here for rough work/calculations */
```

i=1 j=2 i=3

p → k

j = 2 + 3 = 5

k = 11

p → i q → p  
r → k

k = k - \*q  
k = k - 1

3 5

2

Gives an error  
∵ address getting  
subtracted from  
integer and then  
being stored in  
in integer variable.

3. [2 marks] A student wrote following code for reversing an input integer array A of n elements. But on execution, it is observed that the code is wrong. Student approached the TA Raman who replied that there is/are **small mistake(s)** in this code. Spot the mistake(s) (Encircle that line(s)) & mention what should be the correct statement/expression(s) there.

```
void reverse(int A[], int n) {
    int i, j, temp;
    i=0;
    while(i < n){
        j = n-1-i;
        temp = A[i];
        A[i] = A[j];
        A[j] = temp;
        i++;
    }
}
```

Correct

```
void reverse (int A[], int n) {
    int i, j, temp;
    i=0;
    while (i < n/2) {
        j = n-1-i;
        temp = A[i];
        A[i] = A[j];
        A[j] = temp;
        i++;
    }
}
```

4. [3 marks] Refer to following partial C code to transpose a square matrix (or say 2D array). Complete the code ( .... part) without using any additional array and without declaring any additional variable.

```
#include <stdio.h>
#define N 12 /* this value 12 may vary by program user */
void main() {
    int A[N][N]; int i,j,k,temp1,temp2;
    printf("\n Input the NxN matrix elements where N= %d . \n", N);
    for (i=0;i<N;i++) {
        for (j=0;j<N;j++)
            scanf("%d ", &(A[i][j]));
    }
```

```
1 2 3
4 5 6
7 8 9

1 4 7
2 5 8
5 6 9
```

```
....for (i=0; i<N; i++) {
    for (j=0; j<N; j++) {
        temp = A[i][j];
        A[i][j] = A[j][i];
        A[j][i] = temp;
    }
}
```

```
for (i=0; i<N; i++) {
    for (j=0; j<N; j++)
        printf("%d ", A[i][j]);
    printf("\n");
}
```

```
printf("\n Following is the TRANSPOSE matrix \n");
for (i=0; i<N; i++) { printf("\n");
    for (j=0; j<N; j++)
        printf("%d ", (A[i][j]));
}
```





5. [5 marks] Given an input string **inp**, complete the C program below that does the following
- It first computes the total number of those characters that appear twice or more in the input string.
  - Then it removes all digits (if any in the input string) and also changes the input string alphabets to lowercase. Then it prints this modified input string as output string.
- As an example, if input string **inp** is "**Animesh181SharmAaa**", the output would be

**No. of characters that repeat = 5**

**Output String: animeshsharmaaa**

*/\*Ans above 5 because A, m, h, 1, and a are the characters that appear again \*/*

```
#include<stdio.h>
/*you are not permitted to use any other library functions */
#define SZ 1000
void main() {
    int i,j,k,temp1,temp2; char c1, c2, c3;
    char inp[SZ]; scanf("%s", inp);
    // ... Assuming 's' and 'S' are treated as different characters
    int i=0;
    for(i=0; i<=255; i++) {
    for(j=0; j<
    for(i=0; inp[i]!='\0'; i++) {
    temp1=i;
    temp2=0;
    for(i=0; i<=255; i++) {
        k=0;
        for(j=0; j<temp1; j++) {
            if(inp[j] == (char)i) {
                k++;
            }
            if(k>=2)
                temp2++;
        }
    }
    printf("No. of characters that repeat=%d \n", temp2);
    for(i=0; i<temp1; i++) {
        if(inp[i] <='9' && inp[i] >='0')
            continue;
        else {
            if(inp[i] <='z' && inp[i] >='A')
                printf("%c", inp[i]+32);
            else
                printf("%c", inp[i]);
        }
    }
}
```

*checking for all characters*

*4*

(Note: You may safely assume that size of the input string is less than 1000. You may write the code within the **main** function to achieve the purpose or you may write a separate function e.g. **int fun1(char \*arr)** and call that function appropriately within **main** function to achieve the purpose )

6. [7 marks] Consider a singly linked list (based on NODE structure as mentioned below) referred using the global node pointer variable **head**. Write the C code for successfully deleting the (first appearing) node having data value **key**. If there is no node in the linked list that has data value **key**, the code brings no change to the linked list. If there are multiple nodes with data value **key**, the code deletes that one which appears first while traversing the linked list using global pointer variable **head**.

```
typedef struct node{
    int data;
    struct node * next;
} NODE;
```

Function prototype is as follows - **void find\_delete( int key );**

```
#include <stdio.h>
```

```
typedef struct node {
    int data;
    struct node * next;
} NODE;
```

```
NODE * head;
```

```
void find_delete(int key);
```

```
void main() {
    int key;
    printf("Enter the key \n");
    scanf("%d", &key);
    find_delete(key);
}
```

```
void find_delete(int key) {
```

```
    NODE * temp, t1,
```

```
    temp = (NODE *) malloc (sizeof(NODE));
```

```
    t2 = head;
```

```
    t1 = head -> next
```

```
    if (t1 -> data == key) {
        t1 = t1 -> next;
```

```
    if (head == null)
```

```
        printf("NO Element to delete");
```

```
    else {
```

```
        if (head -> next == NULL) {
```

```
            if (head -> data == key) {
```

```
                head -> data = NULL;
```

```
                free(head);
```

```
            }
```

```
        }
```



```
if (t2 -> data != key) {  
    t1 = t1 -> next;  
} temp = t1 -> next;  
t1 -> data = NULL;  
t1 -> next = NULL;  
free(t1);
```

else {

*while* if ((t1 -> data != key) && t1) {

```
t2 = t1;  
t1 = t1 -> next;
```

```
}  
if (t1 == NULL) {  
    temp = t1 -> next;
```

```
t1 -> data = NULL;  
t1 -> next = NULL;  
free(t1);
```

```
t2 -> next = temp;  
temp = NULL;  
t2 = NULL;  
t1 = NULL;
```

}

else

if (t1 == NULL)

{

```
t1 -> data = NULL;  
free(t1);  
t2 -> next = NULL;  
t1 = NULL;
```

}

}

*while loop?*

*Not needed, Not done.*

*Wrong!*

3.5