1. Error Correction and Valid Variable Detection[CO1]

Question 1: Correct the errors in the following code:

```
#include <stdio.h>
void main() {
        int a = 5; b = 6, sum
        if (a > b) {
                a-b =sum;
        } else
                sum = a + b;
        printf("Sum is: %c", sum);
}
```

Question 2: Identify invalid variable names from the following:

_totalAmount, int, calc.total, amount@value, 2ndPlace, __finalResult__

Question 3: Correct the errors in the following code:

```
#include <studio.h>
void main() {
   int a = 5; int b = 10; int c;
   c = a + b,
   if (c > 10) {
      printf("Sum is greater than 10\n");
   } else {
      printf("Sum is less than %d or equal to 10\n");
   }
   return 0;
}
```

Question 4: Identify invalid variable names from the following:

main_value, void, total-value, final_sum, *value, _3value

Question 5: Correct the errors in the following code:

```
include <stdio.h>
int main() {
   int a = A; float b = 5.5; float sum;
   sum = a + b;
   if (sum > 5) {
      printf("Sum: %f is greater than 5", sum);
   } else {
      printf("Sum: %f is less than or equal to 5", sum);

   return 3.5;
}
```

Question 6: Identify invalid variable names from the following:

calc_sum, float, variable-value, _input, sumTotal, 5_sum

Question 7: Correct the errors in the following code:

```
#include {stdio.h}
int main() {
    int a, b = 10, c;a=b;
    a == 5;
    c = a + b;
    for (int j = 0; i < 3; i++) {
        c += i;
    }
    print("%d\n", c);
    return 0;
}
```

Question 8: Identify invalid variable names from the following:

sumValue, while, sum.value, variable*name, 1_value, _variable

Question 9: Correct the errors in the following code:

```
include <stdio.h>
int main() {
    int a = 2; int b = 4; int sum;
    sum = a + b;
    if (sum % 2 = 0) {
        printf("Total: %d is even", Sum);
    } else {
        Printf("Total: %f is odd", sum);
    }
    return o;
}
```

Question 10: Identify invalid variable names from the following:

finalResult, return, var.value, sum-total, 3rd_variable, __sum__

2. If-Else to Switch Conversion and Vice Versa[CO1]
Question 1: Convert the following switch statement to if-else:

```
switch(x) {
    case 1:
        printf("One");
        break;
    case 2:
        printf("Two");
        if (x == 2) {
            printf(" - Second");
        }
        break;
    case 3:
        printf("Three");
        break;
    default:
```

```c
      printf("Other");
}
```

Question 2: Convert the following if-else to switch:

```c
if (y == 1) {
   printf("January");
} else if (y == 2) {
   printf("February");
} else if (y == 3) {
   printf("March");
   if (y == 3) {
      printf(" - Third month");
   }
} else {
   printf("Other");
}
```

Question 3: Convert the following switch statement to if-else:

```c
switch(day) {
   case 1:
      printf("Monday");
      break;
   case 2:
      printf("Tuesday");
      break;
   case 3:
      printf("Wednesday");
      if (day == 3) {
         printf(" - Midweek");
      }
      break;
   case 4:
      printf("Thursday");
      break;
   case 5:
      printf("Friday");
      break;
   case 6:
      printf("Saturday");
      break;
   case 7:
      printf("Sunday");
      break;
   default:
      printf("Invalid");
}
```

Question 4: Convert the following if-else to switch:

```c
if (color == 'r') {
   printf("Red");
} else if (color == 'g') {
   printf("Green");
} else if (color == 'b') {
   printf("Blue");
   if (color == 'b') {
      printf(" - Cool color");
```

```
    }
} else {
    printf("Unknown");
}
```

Question 5: Convert the following switch statement to if-else:

```
switch(choice) {
    case 'a':
        printf("Apple");
        break;
    case 'b':
        printf("Banana");
        break;
    case 'c':
        printf("Cherry");
        if (choice == 'c') {
            printf(" - Sweet fruit");
        }
        break;
    default:
        printf("None");
}
```

Question 6: Convert the following if-else to switch:

```
if (status == 0) {
    printf("Inactive");
} else if (status == 1) {
    printf("Active");
} else if (status == 2) {
    printf("Pending");
    if (status == 2) {
        printf(" - Awaiting approval");
    }
} else {
    printf("Unknown");
}
```

Question 7: Convert the following switch statement to if-else:

```
switch(level) {
    case 1:
        printf("Beginner");
        break;
    case 2:
        printf("Intermediate");
        break;
    case 3:
        printf("Advanced");
        if (level == 3) {
            printf(" - Expert level");
        }
        break;
    default:
        printf("Unknown");
}
```

Question 8: Convert the following if-else to switch:

```c
if (age < 13) {
    printf("Child");
} else if (age < 20) {
    printf("Teen");
} else if (age < 65) {
    printf("Adult");
    if (age >= 18 && age < 65) {
        printf(" - Working age");
    }
} else {
    printf("Senior");
}
```

Question 9: Convert the following switch statement to if-else:

```c
switch(operation) {
    case '+':
        result = a + b;
        break;
    case '-':
        result = a - b;
        break;
    case '*':
        result = a * b;
        if (operation == '*') {
            printf(" - Multiplication");
        }
        break;
    case '/':
        result = a / b;
        break;
    default:
        printf("Invalid Operation");
}
```

Question 10: Convert the following if-else to switch:

```c
if (grade == 'A') {
    printf("Excellent");
} else if (grade == 'B') {
    printf("Good");
} else if (grade == 'C') {
    printf("Average");
    if (grade == 'C') {
        printf(" - Needs improvement");
    }
} else if (grade == 'D') {
    printf("Below Average");
} else {
    printf("Fail");
}
```

3. For Loop to While Conversion and Vice Versa[CO1]
Question 1: Convert the following for loop to a while loop:

```c
for (int i = 0; i < 5; i++) {
    for (int j = 0; j < i; j++) {
        printf("%d ", j);
```

```
    }
    printf("\n");
}
```

Question 2: Convert the following while loop to a for loop:

```
int i = 0;
while (i < 5) {
    int j = 0;
    while (j < i) {
        printf("%d ", j);
        j++;
    }
    printf("\n");
    i++;
}
```

Question 3: Convert the following for loop to a while loop:

```
for (int j = 10; j > 0; j--) {
    for (int k = j; k > 0; k--) {
        printf("%d ", k);
    }
    printf("\n");
}
```

Question 4: Convert the following while loop to a for loop:

```
int j = 10;
while (j > 0) {
    int k = j;
    while (k > 0) {
        printf("%d ", k);
        k--;
    }
    printf("\n");
    j--;
}
```

Question 5: Convert the following for loop to a while loop:

```
for (int k = 0; k <= 20; k += 2) {
    for (int l = k; l >= 0; l -= 2) {
        printf("%d ", l);
    }
    printf("\n");
}
```

Question 6: Convert the following while loop to a for loop:

```
int k = 0;
while (k <= 20) {
    int l = k;
    while (l >= 0) {
        printf("%d ", l);
        l -= 2;
    }
    printf("\n");
    k += 2;
```

```
}
```

Question 7: Convert the following for loop to a while loop:

```
for (int m = 1; m <= 15; m += 3) {
    for (int n = m; n < m + 3; n++) {
        printf("%d ", n);
    }
    printf("\n");
}
```

Question 8: Convert the following while loop to a for loop:

```
int m = 1;
while (m <= 15) {
    int n = m;
    while (n < m + 3) {
        printf("%d ", n);
        n++;
    }
    printf("\n");
    m += 3;
}
```

Question 9: Convert the following for loop to a while loop:

```
for (int n = 0; n < 50; n += 5) {
    for (int o = n; o < n + 5; o++) {
        printf("%d ", o);
    }
    printf("\n");
}
```

Question 10: Convert the following while loop to a for loop:

```
int n = 0;
while (n < 50) {
    int o = n;
    while (o < n + 5) {
        printf("%d ", o);
        o++;
    }
    printf("\n");
    n += 5;
}
```

4. Manual Tracing of 2D Array[CO3]

Question 1: Manually trace the following code and show changes to variables:

```
int arr[2][2] = {{1, 2}, {3, 4}};
for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 2; j++) {
        arr[i][j] += i + j;
    }
}
```

Question 2: Manually trace the following code and show changes to variables:

```
int arr[3][3] = {{1, 0, 0}, {0, 1, 0}, {0, 0, 1}};
```

```
for (int i = 0; i < 3; i++) {
   for (int j = 0; j < 3; j++) {
      arr[i][j] *= 2;
      if (arr[i][j] == 0) {
         arr[i][j] = 1;
      }
   }
}
```

Question 3: Manually trace the following code and show changes to variables:

```
int arr[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
for (int i = 0; i < 3; i++) {
   for (int j = 0; j < 3; j++) {
      if ((i + j) % 2 == 0) {
         arr[i][j] += 1;
      } else {
         arr[i][j] -= 1;
      }
   }
}
```

Question 4: Manually trace the following code and show changes to variables:

```
int arr[2][3] = {{1, 2, 3}, {4, 5, 6}};
for (int i = 0; i < 2; i++) {
   for (int j = 0; j < 3; j++) {
      arr[i][j] = arr[i][j] * (i + 1);
   }
}
```

Question 5: Manually trace the following code and show changes to variables:

```
int arr[3][2] = {{1, 4}, {2, 5}, {3, 6}};
for (int i = 0; i < 3; i++) {
   for (int j = 0; j < 2; j++) {
      arr[i][j] += arr[i][j];
      if (arr[i][j] % 2 == 0) {
         arr[i][j] /= 2;
      }
   }
}
```

Question 6: Manually trace the following code and show changes to variables:

```
int arr[4][4] = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}, {13, 14, 15, 16}};
for (int i = 0; i < 4; i++) {
   for (int j = 0; j < 4; j++) {
      arr[i][j] *= (i + j);
   }
}
```

Question 7: Manually trace the following code and show changes to variables:

```
int arr[3][3] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};
for (int i = 0; i < 3; i++) {
   for (int j = 0; j < 3; j++) {
      if (i == j) {
         arr[i][j] = 0;
```

```
        } else {
            arr[i][j] = 1;
        }
    }
}
```

Question 8: Manually trace the following code and show changes to variables:

```
int arr[2][2] = {{1, 2}, {3, 4}};
for (int i = 0; i < 2; i++) {
    for (int j = 0; j < 2; j++) {
        if (i != j) {
            arr[i][j] = 0;
        }
    }
}
```

Question 9: Manually trace the following code and show changes to variables:

```
int arr[4][4] = {{1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}, {13, 14, 15, 16}};
for (int i = 0; i < 4; i++) {
    for (int j = 0; j < 4; j++) {
        if ((i + j) % 2 == 0) {
            arr[i][j] *= 2;
        } else {
            arr[i][j] /= 2;
        }
    }
}
```

Question 10: Manually trace the following code and show changes to variables:

```
int arr[3][3] = {{2, 4, 6}, {8, 10, 12}, {14, 16, 18}};
for (int i = 0; i < 3; i++) {
    for (int j = 0; j < 3; j++) {
        arr[i][j] += i * j;
    }
}
```

5. Manual Tracing of Nested Loop[CO1]
Question 1: Manually trace the following code and show changes to variables:

```
int total = 0;
for (int i = 1; i <= 3; i++) {
    for (int j = 1; j <= 3; j++) {
        total += i * j;
    }
}
printf("Total: %d", total);
```

Question 2: Manually trace the following code and show changes to variables:

```
int product = 1;
for (int i = 2; i <= 4; i++) {
    for (int j = 2; j <= 4; j++) {
        product *= i + j;
    }
}
printf("Product: %d", product);
```

Question 3: Manually trace the following code and show changes to variables:

```c
int sum = 0;
for (int i = 1; i <= 4; i++) {
   for (int j = 1; j <= i; j++) {
      sum += i - j;
   }
}
printf("Sum: %d", sum);
```

Question 4: Manually trace the following code and show changes to variables:

```c
int count = 0;
for (int i = 1; i <= 5; i++) {
   for (int j = i; j <= 5; j++) {
      count += j;
   }
}
printf("Count: %d", count);
```

Question 5: Manually trace the following code and show changes to variables:

```c
int value = 1;
for (int i = 1; i <= 3; i++) {
   for (int j = 1; j <= 3; j++) {
      value *= i + j;
   }
}
printf("Value: %d", value);
```

Question 6: Manually trace the following code and show changes to variables:

```c
int total = 0;
for (int i = 1; i <= 4; i++) {
   for (int j = 1; j <= 4; j++) {
      total += i * j - 1;
   }
}
printf("Total: %d", total);
```

Question 7: Manually trace the following code and show changes to variables:

```c
int result = 0;
for (int i = 1; i <= 3; i++) {
   for (int j = 1; j <= 3; j++) {
      result += i * j + 2;
   }
}
printf("Result: %d", result);
```

Question 8: Manually trace the following code and show changes to variables:

```c
int outcome = 1;
for (int i = 2; i <= 4; i++) {
   for (int j = 2; j <= 4; j++) {
      outcome *= i - j + 1;
   }
}
```

```
    printf("Outcome: %d", outcome);
```

Question 9: Manually trace the following code and show changes to variables:

```
int sum = 0;
for (int i = 1; i <= 3; i++) {
   for (int j = 1; j <= 3; j++) {
      sum += i + j;
   }
}
printf("Sum: %d", sum);
```

Question 10: Manually trace the following code and show changes to variables:

```
int final = 0;
for (int i = 1; i <= 4; i++) {
   for (int j = 1; j <= i; j++) {
      final += i * j;
   }
}
printf("Final: %d", final);
```

6. Manual Tracing of 1D Array Code Segments[CO3]

Question 1

Manually trace the given code segment below. Show the changes of all the variables i, hi, hlw, and array elements arr in each step.

```
int hi = 0, hlw = 10;
int arr[4] = {10, 20, 30, 40};
for(int i = 4; i <= hlw; i++) {
   arr[hi] = arr[hi + 1] - 5;
   hlw -= 2;
}
```

Question 2

Manually trace the given code segment. Show the changes of all the variables i and array elements ara in each step.

```
int ara[5] = {8, 6, 2, 4, 7};
for(int i = 1; i < 5; i += 2) {
   ara[i] = 3 * ara[i - 1];
}
for(int i = 1; i < 5; i++) {
   if(i % 2 == 0) {
      ara[i] = i * 4 + ara[i - 1];
   }
}
```

Question 3

Manually trace the given code segment. Show the changes of all the variables i, j, jump, and array elements A and B in each step.

```
int A[4] = {3, 2, 1};
int B[4] = {10, 20, 30};
int jump = 100;
for(int i = 0; i < 3; i++) {
    jump = A[i] * 2;
    for(int j = 0; j < 3; j++) {
        B[i] = A[i] + B[i];
        jump = B[i] / 2;
    }
    A[i]++;
}
```

Question 4

Manually trace the given code segment. Show the changes of all the variables i and array elements arr in each step.

```
int arr[6] = {5, 10, 15, 20, 25, 30};
for(int i = 0; i < 6; i++) {
    if(arr[i] % 10 == 0) {
        arr[i] += 5;
    } else {
        arr[i] -= 2;
    }
}
```

Question 5

Manually trace the given code segment. Show the changes of all the variables i and array elements arr in each step.

```
int arr[5] = {1, 2, 3, 4, 5};
for(int i = 0; i < 5; i++) {
    arr[i] = arr[i] * 2;
}
for(int i = 4; i >= 0; i--) {
    arr[i] = arr[i] - 1;
}
```

Question 6

Manually trace the given code segment. Show the changes of all the variables i, j, and array elements A in each step.

```
int A[4] = {1, 2, 3, 4};
for(int i = 0; i < 4; i++) {
    for(int j = i; j < 4; j++) {
        A[i] = A[i] * (j + 1) - A[j];
```

```
    }
}
```

Question 7

Manually trace the given code segment. Show the changes of all the variables i, j, sum, and array elements arr in each step.

```
int arr[5] = {1, 3, 5, 7, 9};
int sum = 0;
for(int i = 0; i < 5; i++) {
   for(int j = i; j < 5; j++) {
      sum += arr[j];
      arr[j] = sum - arr[i];
   }
}
```

Question 8

Manually trace the given code segment. Show the changes of all the variables i, temp, max, and array elements arr in each step.

```
int arr[5] = {2, 4, 6, 8, 10};
int max = arr[0];
for(int i = 0; i < 5; i++) {
   int temp = arr[i];
   if(temp > max) {
      max = temp;
   }
   arr[i] = arr[4 - i];
   arr[4 - i] = temp;
}
```

Question 9

Manually trace the given code segment. Show the changes of all the variables i, j, k, and array elements arr in each step.

```
int arr[6] = {1, 1, 1, 1, 1, 1};
for(int i = 1; i < 6; i++) {
   for(int j = 0; j < i; j++) {
      for(int k = j; k < i; k++) {
         arr[i] += arr[k];
      }
   }
}
```

Question 10

Manually trace the given code segment. Show the changes of all the variables i, sum, product, and array elements arr in each step.

```
int arr[4] = {5, 10, 15, 20};
int sum = 0, product = 1;
for(int i = 0; i < 4; i++) {
    sum += arr[i];
    product *= arr[i];
    arr[i] = sum - product;
}
```

7. Printing Pattern using nested loop
Write code in C language to print following patterns (use nested loops):
Input: n (an integer number saying how many lines will your pattern have)
Output: Printed pattern
#Examples are for n=5

``` 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 ```
 (aligned:)
 `        1`
 `      2 3`
 `    4 5 1`
 `  2 3 4 5`
 `1 2 3 4 5`
	`    *`
	 `    *`
	 `*****`
	 `    *`
	 `    *`

`      1`
 `    2 1 2`
 `  3 2 1 2 3`
 `4 3 2 1 2 3 4`
	`*****`
	 `*    *`
	 `*    *`
	 `*    *`
	 `*****`

`*        *`
 ` *      *`
 `   *`
 ` *    *`
 `*        *`
	`    *`
	 `  ***`
	 `*****`

`*`
 `**`
 `***`
 `****`
 `*****`
 `****`
 `***`
 `**`
 `*`
	`*****`
	 ` ***`
	 `  *`

`  *****`
 `  *****`
 `  *****`
 `  *****`
 `  *****`
	`  *`
	 ` ***`
	 `*****`
	 ` ***`
	 `  *`
```