CTII Set 3 answer key

Programming For Problem Solving (SRM Institute of Science and Technology)

SRM Institute of Science and Technology
College of Engineering and Technology
School of Computing

SET 3

SRM Nagar, Kattankulathur – 603203, Chengalpattu District, Tamil Nadu

Academic Year: 2023-24 (ODD)

Test: CLA-T2 **Date: 3-11-2023**

Course Code & Title: 21CSS101J & Programming for Problem SolvingDuration: 1 hour 40 mins

Year &Sem: I Year / I Sem **Max. Marks:** 50

SET 3
ANSWER KEY

1) Create two arrays with five subject marks each, and combine the marks into a new array so that the new array needs to be arranged as lowest marks to highest marks. Print the new array as the output. Write a C program without applying user-defined functions.

Input Format
Enter the number of elements for First Array : 5
Enter the elements for First Array : 4 13 12 1 10
Enter the number of elements for Second Array :5
Enter the elements for Second Array : 14 6 7 8 9
Output Format
Elements After Merging :1 4 6 7 8 9 10 12 13 14

ANSWER:

```
#include <stdio.h>
int main() {
    // Initializing the arrays
    int array1[5] = {1, 2, 3, 4, 5};
    int array2[5] = {6, 7, 8, 9, 10};
    // Merging the arrays
    int mergedArray[10];
```

```
    int i, j;
    for (i = 0; i < 5; i++) {
        mergedArray[i] = array1[i];
    }
    for (i = 0, j = 9; i < 5; i++, j--) {
        mergedArray[j] = array2[i];
    }
    // Sorting the merged array
    for (i = 0; i < 9; i++) {
        for (j = 0; j < 9 - i; j++) {
            if (mergedArray[j] > mergedArray[j + 1]) {
                int temp = mergedArray[j];
                mergedArray[j] = mergedArray[j + 1];
                mergedArray[j + 1] = temp;
            }
        }
    }
    // Printing the sorted merged array
    printf("Sorted Merged Array: ");
    for (i = 0; i < 10; i++) {
        printf("%d ", mergedArray[i]);
    }
    return 0;
}
```

2) Ten year old Vishwat is engaged in a game where he must swap the two numbers, A and B, using a Function. Vishwat tried to pass the value of A and B as a parameter of the function. But, he did not get the correct output. So, you please help him to swap A and B by writing C programs using two different ways of argument passing methods.

Input Format

Input line contains two numbers A and B.

Output Format

Output in a single line the two numbers after swapping them.

ANSWER:

```
#include <stdio.h>
// Function to swap two numbers
void swap_numbers(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}
int main() {
```

```
    int A, B;
        // Getting input from the user
    printf("Enter the first number (A): ");
    scanf("%d", &A);
    printf("Enter the second number (B): ");
    scanf("%d", &B);
    // Displaying the numbers before swapping
    printf("Before swapping: A = %d, B = %d\n", A, B);
    // Swapping the numbers
    swap_numbers(&A, &B);
    // Displaying the numbers after swapping
    printf("After swapping: A = %d, B = %d\n", A, B);
    return 0;
}
```

3) Sharmila and Meena are playing a word game. In the game, the opponent has to pick a word S. If the word contains more than 2 vowels, then it is declared as the winner. Find the winner using the C program.

Sample Input and Output

Input: hihello Output: Won

Input: axpyfv Output: Lost

ANSWER

```
#include <stdio.h>
#include <string.h>
int main() {
   char word[100];
   printf("Enter a word: ");
   scanf("%s", word);
   int vowels = 0;
   for (int i = 0; i < strlen(word); i++) {
      if (word[i] == 'a' || word[i] == 'e' || word[i] == 'i' || word[i] == 'o' || word[i]
== 'u') {
         vowels++;
      }
   }
   if (vowels > 2) {
      printf("Won\n");
   } else {
      printf("Lost\n");
   }
   return 0;
}
```

4) Write a C program using the functionvoid climb(intstep_arr[], int reachable[]); which reads the number of steps taken by a person on each jump while climbing a staircase in an array. Compute and print all the step numbers traversed from ground level (step 0).

Input:
No. of jumps:
6
Number of steps in each jump:
2 1 3 1 2 2
Step Numbers Traversed:
2 3 6 7 9 11
ANSWER

```
#include <stdio.h>

void climb(int step_arr[], int reachable[]) {
    int i, sum = 0;
    reachable[0] = 1;

    for (i = 0; step_arr[i] != 0; i++) {
        sum += step_arr[i];
        reachable[sum] = 1;
    }
}

int main() {
    int step_arr[100];
    int reachable[100] = {0};
    int i = 0, j;

    printf("Enter the number of steps taken by the person on each jump (0
to stop): \n");
    while (1) {
        scanf("%d", &step_arr[i]);
        if (step_arr[i] == 0) {
            break;
        }
        i++;
    }

    climb(step_arr, reachable);

    printf("\nStep numbers traversed from the ground level are: \n");
    for (j = 0; reachable[j] != 0; j++) {
```

```
            printf("%d ", j);
        }

        return 0;
}
```

5) The Professor needs to calculate the average marks of a student, and he wants to display the grade obtained by the student.

If the student scores an average greater than 75, then the grade is Distinction.
If average is 60>= and <75, then the grade is First Division.
If average is 50 >= and <60, then the grade is Second Division.
If average is 40>= and <50, then the grade is Third Division.
Else the grade is Fail using the C user-defined function.
Mandatory Test Case:
void avg_marks(int marks[]);
void grade(float avg);
Sample Input:
Enter Marks for each subject
Sample Output:
Display the total, average and grade

ANSWER

```
    def calculate_grade():
        # Assuming marks are out of 100 for simplicity
        marks = []
        for i in range(5):
            marks.append(float(input(f"Enter marks of subject {i+1}: ")))

        total = sum(marks)
        aggregate = total / 5

        if aggregate > 75:
            grade = "Distinction"
        elif 60 <= aggregate < 75:
            grade = "First Division"
        elif 50 <= aggregate < 60:
            grade = "Second Division"
        elif 40 <= aggregate < 50:
            grade = "Third Division"
        else:
            grade = "Fail"

        print(f"Total Marks: {total}")
        print(f"Aggregate: {aggregate}%")
        print(f"Grade: {grade}")
```

```
    # Calling the function to calculate the grade
    calculate_grade()
```

6) Mrs. Pinky is interested in Python function concepts, and she wanted to find out the maximum, minimum of some values, add new element to the list, and remove the duplication of the element using a function.

Input Format

Enter the number of elements for the list: 5

Enter the elements for list: 4 13 12 1 10 1

Output Format

Maximum value: 13

Minimum value: 1

updated list: 4 13 12 1 10 1 8

Duplication Removed: 4 13 12 1 10 8

ANSWER:

```
def find_maximum(elements):
    max_value = elements[0]
    for element in elements:
        if element > max_value:
            max_value = element
    return max_value


def find_minimum(elements):
    min_value = elements[0]
    for element in elements:
        if element < min_value:
            min_value = element
    return min_value


def add_element(elements, new_element):
    elements.append(new_element)


def remove_duplicates(elements):
    unique_elements = []
    for element in elements:
```

```
        if element not in unique_elements:
            unique_elements.append(element)
    return unique_elements


def perform_operations():
    num_elements = int(input("Enter the number of elements for the list: "))
    elements = [int(x) for x in input("Enter the elements for the list: ").split()]

    print(f"Maximum value: {find_maximum(elements)}")
    print(f"Minimum value: {find_minimum(elements)}")

    add_element(elements, 8)  # Adding 8 to the list
    print(f"Updated list: {' '.join([str(x) for x in elements])}")

    elements = remove_duplicates(elements)  # Removing duplicates
    print(f"Duplication Removed: {' '.join([str(x) for x in elements])}")


# Call the function to perform the operations
perform_operations()
```

7) You are developing a student management system for a school. The system needs to store student information: name, register number, DOB, grade, and attendance details. To do this, use the Python dictionary concept.

a)Delete a particular student from the dictionary

b) Modify the name of an existing student

c) Check if a student is present in the dictionary or not

d) Display the dictionary in sorted order of names

ANSWER:

```
# Initializing an empty dictionary to store student information
student_dict = {}

# Adding sample data to the dictionary
student_dict = {
    1234: {"name": "John Doe", "DOB": "2005-05-15", "grade": 7, "attendance": 95},
    5678: {"name": "Jane Smith", "DOB": "2006-07-20", "grade": 8, "attendance": 92},
    9101: {"name": "Michael Johnson", "DOB": "2005-11-30", "grade": 7, "attendance": 98}
}

# a) Delete a particular student from the dictionary
```

```
def delete_student(register_number):
    if register_number in student_dict:
        del student_dict[register_number]
        print(f"Student with register number {register_number} has been deleted.")
    else:
        print(f"Student with register number {register_number} not found.")

# b) Modify the name of an existing student
def modify_name(register_number, new_name):
    if register_number in student_dict:
        student_dict[register_number]["name"] = new_name
        print(f"Name of student with register number {register_number} has been modified to {new_name}.")
    else:
        print(f"Student with register number {register_number} not found.")

# c) Check if a student is present in the dictionary or not
def check_student_presence(register_number):
    if register_number in student_dict:
        print(f"Student with register number {register_number} found in the dictionary.")
    else:
        print(f"Student with register number {register_number} not found in the dictionary.")

# d) Display the dictionary in sorted order of names
def display_sorted_dict():
    sorted_student_dict = dict(sorted(student_dict.items(), key=lambda item: item[1]["name"]))
    print(sorted_student_dict)

# Test the functions
print("Initial Dictionary:")
print(student_dict)

delete_student(5678)
modify_name(1234, "John Doe Jr.")
check_student_presence(9101)
display_sorted_dict()
```