Daffodil International University

Faculty of Science & Information Technology
Department of Computer Science & Engineering
Final Semester Examination, Fall 2024
Course Code: CSE113, Course Title: Programming and Problem Solving
Level: 1 Term: 1 Batch: 67

Time: 2:00 Hrs **Marks: 40**

Answer ALL Questions [Optional]

[The figures in the right margin indicate the full marks and corresponding course outcomes. All portions of each question must be answered sequentially.]

1. **Demonstrate error finding and bug fixing:** Identify the errors in the following code? Explain the errors and reasons why you think they are errors. — [3] — CO2

a)

```
a) #include<stdio.h>
int main() {
  int num = 10;
  int *ptr;
  *ptr = num;
  int num = 27;
  printf("Value of num using pointer: %d\n", *num);
  int arr[3] = {1, 2, 3};
  printf("Array third value: %d\n", arr[3]);
  printf("Pointer address in Hexadecimal: %d\n", ptr);
  return 9.6;
}
```

b) Rewrite the code without any errors. — [3]

2. **Generate** the output of given codes below (write only the output segment in a box): — [3] + [3] — CO3

a)

```
#include<stdio.h>
void printPattern(int n) {
   int i, j;
   for (i = n; i >= 1; i -= 2) {
      for (j = 1; j <= i; j += 2) {
         printf("%d ", j);
      }
      printf("\n");
   }
}
int main() {
   printPattern(5);
   return 0;
}
```

b)

```
#include<stdio.h>
void adjustValue(int *ptr) {
   *ptr += 5;
}
void swapValues(int *ptr1, int *ptr2) {
   int temp = *ptr1;
   *ptr1 = *ptr2;
   *ptr2 = temp;
}
void modifyValues(int *p1, int *p2) {
   adjustValue(p1);
   *p2 -= 10;
   swapValues(p1, p2);
}
int main() {
   int a = 15, b = 20;
   int *p1 = &a;
   int *p2 = &b;
   modifyValues(p1, p2);
   printf("Value of a after modification: %d\n", a);
   printf("Value pointed by p2 (b) after: %d\n", *p2);
   adjustValue(p1);
   printf("Final value pointed by p1: %d\n", *p1);
   return 0; }
```

c)

```
#include<stdio.h>
struct Student {
    int id;
    float grade;
    struct Student *next;
};
void modifyStudent(struct Student *s) {
    s->id = 200;
    s->grade += 5.0;
}
void printStudent(struct Student *s) {
    printf("Student ID: %d\n", s->id);
    printf("Student Grade: %.1f\n", s->grade);
}
int main() {
    struct Student s1 = {101, 85.5, NULL};
    struct Student s2 = {102, 90.0, NULL};
    s1.next = &s2;
    modifyStudent(s1.next);
    s1.grade -= 10;
    printf("Details of Student 1:\n");
    printStudent(&s1);
    printf("\nDetails of Student 2 (modified via Student 1's next pointer):\n");
    printStudent(s1.next);
    return 0;
}
```

[3]

3. Identify the problems scenarios given below to write a full program for each of the following: CO4

a) You have created a time machine and want to secure it with a password made of lowercase letters. The password is considered "**Weak Password**" if it contains the substring "**computerscience**," and "**Strong Password**" otherwise. Do not use any built-in function for string comparison (like **strstr()**). Implement your own logic to compare the string or search for the substring. [5]

Input: A string consisting of only lowercase letters.
Output: A message indicating whether the password is "**strong password**" or "**weak password**" without quote.

Sample Input	Sample Output
weakpassword	Strong Password

Sample Input	Sample Output
strongcomputersciencepassword	Weak Password

b) Create a program to determine the eligibility of students applying for the CSE department at Daffodil International University based on their **SSC** and **HSC GPA** scores. The eligibility criteria are that the total **GPA** of **SSC** and **HSC** exams combined must be at least **9.70**, and neither the **SSC GPA** nor the **HSC GPA** should be less than **4.75**. Define a structure named Student with fields for **first_name**, **last_name**, **ssc_gpa**, **hsc_gpa**, and **phone_no**. The program should read an integer **n** representing the number of students, followed by an array of **n** Student structures containing the details of each student. It should then check each student's eligibility based on the given criteria and print the names (**first name** and **last name**) of the students who meet the eligibility criteria. [5]

Input: First line contains an integer **N** representing the number of students. Following **N** lines contains information about students.
Output: Print the **last_name** and **first_name** of the students who meet the eligibility criteria.

Sample Input	Sample Output
3 Hasan Galiv 5.00 4.94 01513218141 Arafat Alom 5.00 4.70 01813618945 Hasan Jafran 4.84 4.89 01716278543	Galiv Hasan Jafran Hasan

c) At Daffodil International University, a minimum of three quiz exams are conducted for every course to assess students' performance. You are given information about **n** students, where each student has participated in **m** quizzes. Your task is to write a C program that takes the quiz marks of all **n** students and calculates the average quiz marks for each student. The program should accept inputs in a structured format and output the average marks clearly. [5]

Input: The first line contains two integers, **n** and **m**, where **n** represents the number of students and **m** represents the number of exams each student has participated in.
Output: Print **n** lines, where each line contains the average marks of the respective student.

Sample Input	Sample Output
4 5 12 14 13 15 11 15 14 12 13 10 11 13 14 12 13 15 12 13 13 15	13.0 12.8 12.6 13.6

d) Having calculated the average marks of each student in their quiz exams, it is now time to organize the students based on their performance. Your task is to write a C program that accepts the average marks of **n** students and arranges them in ascending order. The program should only display the sorted average marks. [5]

Input: The first line contains a single integer **n**, representing the number of students. The second line contains **n** space-separated numbers, which represent the average marks of the students.
Output: The output should contain the students' average marks organized in ascending order, printed as a single line of space-separated numbers.

Sample Input	Sample Output
5 13.0 12.6 12.8 13.6 11.4	11.4 12.6 12.8 13.0 13.6

e) Imagine one day you will become a hacker and create a virus for fun that changes all the consonant in your friend's text to the # character. Before making the actual virus, you need to write a C program that performs this task on a given text to prove you are capable of becoming. [5]

Input: A string **S** containing the text. The string may contain lowercase alphabets, digits, space and punctuation marks.
Output: The modified string with all consonants replaced by #.

Sample Input	Sample Output
i will be a hacker!	i #i## #e a #a##e#!