38.5/40

S.No. 303850

SHIV NADAR

INSTITUTION OF EMINENCE DEEMED TO BE UNIVERSITY

DELHI NCR

TEST ANSWER BOOK

Name ADYA SINGHAL

Roll No. 2410110027 Major BTech CSE

Course Code CSD101 Course Title Introduction to Computing & Programming

Date 1/10/24 No. of Continuation Sheets Used 0

INSTRUCTIONS FOR THE STUDENTS

1. It is mandatory to fill all the entries on the answer sheet such as Roll No., Name, Course Code etc.
2. Any identification mark at any other place inside the answer sheet will make it liable to be cancelled.
3. Students should take their seats at least ten minutes before the commencement of the exam. Student will not be allowed to leave the examination hall prior to 30 minutes after the commencement of the examination. Candidates arriving late will not be permitted to enter the examination hall 15 minutes after the schedule commencement of the examination.
4. Carrying the Identity Card is mandatory, failing which the student will not be permitted to appear for the examination. The student is required to sign in the space provided for signature on the attendance sheet in the presence of the invigilator in Examination Hall.
5. Student should follow the instructions given by invigilator at all the stages of the examination. Violation may lead to disciplinary action.
6. Any textual material, written or printed, notebooks, notes, programmable calculators, pagers, mobile phones or other electronic devices are not allowed inside the examination hall. Any student who is found either copying or receiving or giving assistance shall be disqualified.
7. Please handle your answer sheet with care. After the exam is over, the answer sheet must be handed over to the invigilator before leaving the exam hall. Any candidate who does not return the answer sheet or is found to take it out side the examination hall will be disqualified from the examand the Proctoral Board may take further action against him/her as per University rules.
8. This answer book contains 12 pages.

Signature of the Student

Signature of the Invigilator

Q.No.	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.	12.	Sub Total	TOTAL
Marks														
Q.No.	13.	14.	15.	16.	17.	18.	19.	20.	21.	22.	23.	24.	Sub Total	
Marks														

Signature of the Examiner

SECTION-A

Ans1) Identifier is a name given to a function / variable or any user defined item in the program.

Rule 1: It cannot have spaces in between.

Eg. new num ✗ newnum ✓

Rule 2: It can only start with an alphabet ~~and~~ or underscore and not with a digit.

Eg. 9alpha ✗ _alpha ✓ OR alpha_ ✓

3

Ans2) We use rules of precedence.

```
7%7+ 7/7 - 7* 7 >>1
0 +1 - 49 >>1
   -48 >>1
```

a>>1 is given by a = a/(2^1)

= -48/(2^1)

= -24

Answer

2

SECTION-A

Ans1) Identifier is a name given to a function / variable or any user defined item in the program.

Rule 1: It cannot have spaces in between.

Eg. new num ✗ newnum ✓

Rule 2: It can only start with an alphabet ~~and~~ or underscore and not with a digit.

Eg. 9alpha ✗ _alpha ✓ OR alpha_ ✓

3

Ans2) We use rules of precedence.

```
7%7+7/7-7*7>>1
0+1-49>>1
   -48>>1
```

a>>1 is given by a = a/(2^1)

= -48/(2^1)

= -24

Answer

2

Ans 3) 3.1

(Ans) (B) b, a ✓ +2

This is because a++ is post increment.

∴ b = 6

and a = a++ = 7

3.2

(Ans) (C) i=0; i<4; i++ +2

The code given is for finding the minimum number in an array.
The minimum number in this array is 8, whereas we get the output as 10, which means the loop does not iterate through the complete array, it only goes till index 3.

Ans 4) 4.1

Dry run :-

i = 0 count = 0

i = 1 count = 1 = (0+1)

i = 2 count = 3 = (1+2)

i = 3 count = 6 = (3+3) ∴ Output is (A) 10

i = 4 count = 10 = (6+4)

4.2

~~Dry Run:-~~ $c = a + b = 6$

Here, the function we have to evaluate is foo(6, 3, 3);

It is equivalent to foo(int *x, int *y, int *z)

$a = a + 1;$ $\Rightarrow a = 4$

$a = c + c =$ ~~$a + b + a + b$~~ $\Rightarrow a = 6 + 6 = 12$

(Ans) (B) 12

4.3

32 bit = 4 byte

sizeof arr = $4 \times 5 = 20$

$\therefore$ arr = 20

arr + 1 = 21

arr + 3 = 23

$\therefore$ Output will be 20 21 23

Ans5) find (a,b) with return 0 if b>a and return a-b is a>b.

Case I: b>a (returns 0)
find (a, 0)

Here a is non negative (can be 0 also)

if a=0 returns 0 and if a>0 it returns a

(a ≯ 0)

0-0=0

Case II: a>b (returns a-b)
find (a, a-b)

Here, a > a-b ∴ it will return a-(a-b) = b

∴ the function returns (D) minimum of a, b.

3.5

Ans6) (C) and (D) correctly satisfy the flow chart.

in (C), acc. to the flow chart if c>d is false then it should print b=2, and condition given in code is if c<=d (implies) then it prints b=2 which means the same thing.

3.5

in (D), all conditions of loop are similar to that given in the flow chart.

SECTION-B

Ans)

```
int addition (int num1, int num2) {
    int sum = num1 + num2;
    return sum;
}

int main() {
int var1;
int var2;
scanf("%d", &var1);
scanf("%d", &var2);
int ans = addition(var1, var2);
printf("The answer is %d", ans);

}
```

function name
function header
return type
parameters
function body
function definition
arguments
function call

In this execution of code, first value of var1 & var2 is taken and then function addition is called with var1 and var2 as arguments.

As soon as the function is called, the compiler reads the function definition and returns the value (here, 'sum') and stores it in a variable 'ans' and prints it.

Ans2)

```
int main() {
int n, id, quan;
int sum=0;
inv_list[5][2] = {{3,10}, {5,30}, {9,12}, {11,15}, {15,80}};


printf("Enter the no. of bought items: ");
scanf("%d", &n);

for (int i=0; i<n; i++){
printf("Enter item id: ");
scanf("%d", &id);
printf("Enter item quantity: ");
scanf("%d", &quan);

for (int j=0; j<5; j++){

  if (id == inv_list[j][0]){
    sum = sum + (quan * inv_list[j][1]);
  }
  }
  }
  }
printf("Total shopping cost is %d", sum);

}
```

+8

Ans2)

```
int main () {

int rows = 4;


for (int i=0; i<rows; i++){

for (int j=0; j < i+1 ; j++){
printf("*");
}
for (int k=0; k < 2*(rows-i-1) ; k++) {
printf(" ");
}
for (int l = 0; l<i+1 ; l++) {
printf ("*" );
}
}


for (int m=rows-1 ; m<=0 ; m--) {
for (int n=0 ; n<m+1 ; n++) {
printf("*");
}
for (int p=0 ; p< 2*(rows-m-1); p++) {
printf(" ");
}
for (int q=0 ; q < m+1 ; q++) {
printf("*");
}
}
}
```