Indian Statistical Institute

BSDS, First Year, Mid-Sem of First Semester Examination, 2025-26

Introduction to Computing

Full Marks: 30 Date: 08-10-2025 Time: 2 Hours

Answer any *three* of the following questions

$3 \times 10 = 30$

1. Consider the following Python program for which each statement is labeled with a unique number from 1 to 7.

1	`n = 15`
2	`n >>= 2`
3	`n = -~n`
4	`n = n<<3 + n<<1`
5	`n **= 2`
6	`n %= n`
7	`print("n =", n)`

For what ordering of the statements numbered 2 to 6, the following outputs can be obtained from the reconstructed programs? Note that no statement is to be skipped, no statement is to be repeated, and the statements numbered 1 and 7 are *not* to be shuffled and will remain in their respective positions. Justify your answers.

(i) `n = 0`, (ii) `n = 1`, (iii) `n = 32`, (iv) `n = 64`, (v) `n = 256`, (vi) `n = 1024`

4+2+1+1+1+1

2. Prove or disprove the following statements:
   (a) The operation `n & ~(1 << k)` will turn the `k`-th bit in the integer `n` to 0.
   (b) The operation `n | (1 << k)` will turn the `k`-th bit in the integer `n` to 1.
   (c) The binary number `[xyx] … 2k times` is always divisible by 11, where `x` and `y` are any arbitrary bit (0 or 1) and `k > 0` is a nonnegative integer.
   (d) The two expressions `n & 1 == 0` and `n & 1 == 1` will get separately evaluated as `True` if the integer `n` is even and odd, respectively.
   (e) The following Boolean expression gets always evaluated as `True`:
   `(A and (not B)) or ((not A) and B) or (not (A or B)) or (A and B).`

2+2+2+2+2

3. (a) Recall that a phrase is *palindrome* if the reverse of the phrase, ignoring the punctuations and blank spaces, is the same as the original phrase given. Prepare a *flowchart* to verify whether a given phrase is palindrome or not. For an example, consider the following sample I/O.

   Sample Input:
   `never odd or even`
   Sample Output:
   `Palindrome`

(b) Given a list `ls` having `n` elements and an index `i ∈ {0, …, n − 1}` as arguments, write a function `NearestLarger(ls, i)` to return the nearest larger value for the number at index `i`. For an example, consider the following sample test run.

Sample Run with Input:

```
ls = [9, 8, 5, 7, 2, 4, 3]
i = 3
print (NearestLarger(ls, i))
```

Sample Output:

```
8
```

Note that the nearest larger element of 7 (present at index 3) is 8.

4+6

4. (a) The following incomplete Python program aims to check whether a number is prime or not and return `True` and `False`, respectively. Complete it by filling up the blank portion and justify your answer.

```
def PrimeCheck(n):
    if n < 2:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            _____
    return True
print(PrimeCheck(29))
```

(b) The following incomplete Python program aims to reverse the digits of a number. For example, for the input `n = 512`, the output will be `215`, but for the input `n = 100`, the output will be `1`. Complete it by filling up the blank portion and justify your answer.

```
n = int(input("Enter a number: "))
rev = 0
while n > 0:
    digit = n % 10
    _____
    n //= 10
print("The reverse of n is:", rev)
```

(c) The following incomplete Python program aims to find out the first non-repeating character in a given string. Complete it by filling up the blank portion and justify your answer.

```
str = "baboon"
found = None
for i in range(len(str)):
    if str.count(str[i]) == 1:
        _____
        break
if found:
    print("First non-repeating character is:", found)
else:
    print("No unique character")
```

3+3+4