Santa Clara University
School of Engineering

Honor Code of the School of Engineering

"All students taking courses in the School of Engineering agree, individually and collectively, that they will not give or receive unpermitted aid in examinations or other course work that is to be used by the instructor as the basis of grading."
-From the Graduate/Undergraduate Bulletin

I have read, understood, and agree to abide by the Honor Code of the School of Engineering.

Name: ____________________________

ID: ____________________________

Signature: ____________________________

Date: ______________________________

1. [30 points]
2. [30 points]
3. [30 points]
4. [30 points]
5. [30 points]
6. [30 points]
7. [30 points]
8. [30 points]
9. [30 points]
10. [30 points]

Total Score:

Midterm Examination

COEN 256 Principles of Programming Languages
Department of Computer Science and Engineering
Santa Clara University

Dr. Ming-Hwa Wang — Fall Quarter 2021
Phone: (408) 805-4175 — Email address: mwang2@cse.scu.edu
Course website: http://www.cse.scu.edu/~mwang2/language/
Office Hours: Friday 9:00pm-9:30pm

1. [30 points] True or false problems:
   a) An interpreter needs more compiling time than a compiler before running/executing.
   b) Python variables have to be defined before reading from them.
   c) If you have *n* strings where *k* (*k* < *n*) of them are duplicates, then Python only keeps *k* – *n* strings instead of *n* strings.
   d) We can put a keyword argument before a non-keyword argument when we call a function.
   e) If arr = [‘2’, ‘10’, ‘4’, ‘8’, ‘1’] then arr.sort() will generate [‘1’, ‘2’, ‘4’, ‘8’, ‘10’].
   f) We use Python list append() and pop() for stack’s push and pop operation, respectively, and use collections.deque’s append() and shift() for queue’s enqueue and dequeue operation, respectively.

2. [30 points] Please write the output of the following Python program:

```
a = [‘a’,’b’,’c’,’d’,’e’,’f’,’g’,’h’]
b = a[4:]
print('Before:   ', b)
b[1] = 99
print('After:    ', b)
print('No change:', a)
```

3. [30 points] Please write the output of the following Python code:

```
def n_length(lst, n):
    if n == 0:
        return [[]]
    l =[]
    for i in range(0, len(lst)):
        m = lst[i]
        remLst = lst[i + 1:]
        for p in n_length(remLst, n-1):
            l.append([m]+p)
    return l
arr ="abc"
print(n_length([x for x in arr], 2))
```

4. [30 points] What is the output of the following program?

```
import re
pat = \
re.compile(r"(?P<quote>['\"])(?P<string>.*?)(?<!\\)(?P=quote)")
def foo(s):
    m = pat.search(s)
    if m:
        return m.group('string')
print(foo('"John","Mary"'))
```

5. [30 points] Please write single line of code to generate each of the following output: a) generate a list of integer squares from 1 to *n* using list comprehension e.g., [1, 4, 9, 16, … 100] if *n* == 10, b) generate last digit in character of all elements in the list in a) using list comprehension, e.g., ['1', '4', '9', '6', '5', '6', '9', '4', '1', '0'] and c) generate all unique characters from b) in any random order, e.g., ['1', '9', '0', '4', '5', '6'].

6. [30 points] Show how to get the big-Oh for the following recursion equations assume $n = 2^k$: T(1) = 1, T(n) = 2T(n/2) + 1

7. [30 points] Please write the output of the following program, and show how to get the big-Oh for the program:

```
def foo(n):
  s = 0
  for i in range(n, 0, -1):
    for j in range(i, 0, -1):
      if (j % i == 0):
        for k in range(j, 0, -1):
          s += 1
  return s
print(foo(2))
print(foo(4))
print(foo(8))
```

8. [30 points] Please fix the 3 bugs in the following Python program which put anagrams together as output.

```
def canonical(s):
  arr = list(s)
  arr.sort()
  return join(arr)
ht = {}
a = ["cba", "yxz", "acb", “xyz”]
for s in a:
  ht[canonical(s)].append(s)
a = []
for k in ht:
  a.extend(ht[k])
print(a)
```

9. [30 points] Given the code below (with 3 bugs inside deep_copy_dict()) which do a deep copy of nested dict and make all values negative intentionally to show the code is working:

```
def deep_copy_dict(a):
  t = {}
  for i in a:
    if isinstance(i, dict):
      t[i] = deep_copy_dict(i)
    else:
      t[i] = -i
  return t
a = {
  'a':3, 'i':{'b':1, 'c':2},
  'j':{'k':{'d':3, 'e':4}, 'l':{'f':5, 'g':6, 'h':7}}
}
b = deep_copy_dict(a)
print(b)
```

The output is:

```
{'a': -3, 'i': {'b': -1, 'c': -2}, 'j': {'k': {'d': -3, 'e': -4},
'l': {'f': -5, 'g': -6, 'h': -7}}}
```

10. [30 points] Please implement factorization code and generate the output as below:

```
>>> def factorization(p, x, b):
...   pass
>>> primes = []
>>> factorization(primes, 220, 2)
>>> print(primes)
[2, 2, 5, 11]
```