Answer: Answers, inline, preceded by red boxes. See exam for full questions and formatting.

Mock Final Exam
CSci 127: Introduction to Computer Science
Hunter College, City University of New York

Spring 2025

1. (a) What will the following Python code print:

```
s = "SundayMondayTuesdayWednesdayThursdayFridaySaturday!!!"
num = s.count("day")
days = s.split("day")
print("There are", num, "days.")
print("Last element is", days[-1])
mess = days[0]
print("Weekends", mess, "and", days[-2])
short = [day[:3] for day in days[:-1]]
print("Weekdays:", short[1:6])
```

Answer:

```
There are 7 days.
Last element is !!!
Weekends Sun and Satur
Weekdays: ['Mon', 'Tue', 'Wed', 'Thu', 'Fri']
```

(b) Consider the following shell commands:

```
$ ls
hello.cpp        photos        pp_hello.py         work
$ pwd
/Users/csguest
```

Assuming the commands below are run sequentially, what is the output after each has run:

i.
```
$ mv hello.cpp   p1.cpp
$ ls
```

Answer:

```
p1.cpp        photos        pp_hello.py        work
```

ii.
```
$ mkdir cprogs
$ mv *.cpp  cprogs
$ ls
```

Answer:

iii.
```
cprogs  photos  pp_hello.py  work
$ cd cprogs
$ pwd
```

Answer:

```
/Users/csguest/cprogs
```

iv.
```
$ mkdir p50_60
$ mkdir pp_5
$ ls | grep pp
```

Answer:

```
p1.cpp  pp_5
```

2. (a) Fill in the missing values in the table:

Answer:

Decimal	Binary	Hexadecimal
3	**11**	3
11	1011	B
34	100010	**22**
254	11111110	**FE**

(b) Fill in the missing information to make the statements true:

Answer:

```
import turtle
megan = turtle.Turtle()
megan.color("#AAAAAA")
ben = turtle.Turtle()
turtle.colormode(1.0)
ben.color(0, 0.75, 0)
ben.color(0, 0.75, 0.0)
seth = turtle.Turtle()
turtle.colormode(255)
seth.color(200, 0, 200)
daniel = turtle.Turtle()
daniel.color("#FF0000")
blake = turtle.Turtle()
turtle.colormode(255)
blake.color(100, 0, 0)
```

i. daniel is red.

ii. **seth** is purple.

iii. ben is green.

iv. **megan** is gray.

v. **blake** is pink.

(c) Consider the code:

Answer:

```
(i)  1  bin_string = input("Enter a binary number: )
     2  dec_num = 0
     3  for c in bin_string:
     4      dec_num = dec_num * 2
(ii) 5      if c == '1':
     6          dec_num = dec_num + 1
     7  print(dec_num)
```

The answer should include:

- Mark line 1 with a "(i)".
- At end of line 1, should circle the space/parenthesis at the end of the line (where the missing quote should be).
- Mark line 5 with a "(ii)".
- At the end of line 5, should circle the space/parenthesis at the end of the line (where the missing colon should be).

i. **Circle** the code above and mark line with **(i)** that caused this error:

```
line 1: bin_string = input("Enter a binary number: )
                                                   ^
SyntaxError: unterminated string literal (detected at line 1)
```

Write the code that would fix the error:

Answer:

```
bin_string = input("Enter a binary number:")
```

ii. **Box** the code above and mark line with **(ii)** that caused this error:

```
line 5: if c == '1'
                    ^
SyntaxError: expected ':'
```

Write the code that would fix the error:

Answer:

```
if c == '1':
```

3. (a) What is the value (True/False) of `out`:

i.
```
in1 = False
in2 = True
out = in1 or in2
```

Answer:

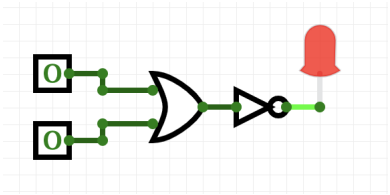
```
out = True
```

ii.
```
in1 = True
in2 = False
out = not in2 and (in2 or not in1)
```

Answer:

```
out = False
```

iii.




```
in1 = False
in2 = False
```

Answer:

```
out = True
```

(b) Fill in the values to yield the output:

i.

in1 = **Answer:** False

in2 = **Answer:** True

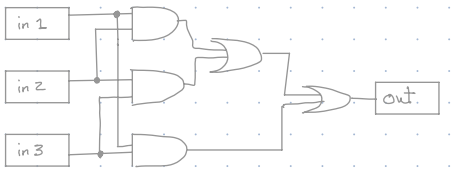
out = True

```
out = not in1 and (in1 or in2)
```

(c) Design a circuit that implements the logical expression:

```
(in1 and in2) or (in1 and in3) or (in2 and in3)
```

Answer:



4. (a) Draw the output for the function calls:

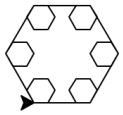
i. `ramble(tia,20,False)` ii. `ramble(tia,40,True)`

```
import turtle
tia = turtle.Turtle()

def ramble(t, len, isNested):
  if len >= 10:
    for i in range(6):
      t.forward(len)
      t.left(60)
      if isNested:
        ramble(t,len/4,isNested)
```

Answer:

Answer:



(b) Given the function definition:

```
def sorted(ls):
    for i in range(4):
        print(ls[0],ls[1],ls[2],ls[3])
        for j in range(3):
            if ls[j] > ls[j+1]:
                ls[j],ls[j+1] = ls[j+1],ls[j]
```

i. What are the formal parameters of `sorted()`? **Answer:** `ls`

ii. What is the return value of `sorted()`? **Answer:** Nothing is returned.

iii. What is the output for `sorted([20,10,0,5])`?

Answer:

ls[0]	ls[1]	ls[2]	ls[3]
20	10	0	5
10	0	5	20
0	5	10	20
0	5	10	20

iv. What is the output for `sorted(["Isabel","Makiya","Georgina","Calvin"])`?

Answer:

ls[0]	ls[1]	ls[2]	ls[3]
Isabel	Makiya	Georgina	Calvin
Isabel	Georgina	Calvin	Makiya
Georgina	Calvin	Isabel	Makiya
Calvin	Georgina	Isabel	Makiya

5. Design an algorithm that finds the finds the highest point from inputted elevation data. Your should ask the user for the name of a file containing a grid of numbers, corresponding to heights above sea level and loads it into a 2D array of integers. The algorithm should find the index `(row, col)` of the **maximum** number in the array.

Answer:

Libraries:	`numpy`
Input:	file with elevation data
Output:	location of highest point (row, col)

Design Pattern:

Answer: ☐ Accumulator ✓ Max/Min ☐ Finding Duplicates ☐ Searching

Principal Mechanisms (select all that apply):

Answer: ☐ Single Loop ✓ Nested Loop ✓ Conditional (if/else) ☐ Recursion
✓ Indexing/slicing ☐ Dictionary ☐ List Comprehension ☐ Regular Expressions

Process (as a concise and precise LIST OF STEPS / pseudocode):

(Assume libraries have already been imported.)

Answer:

(a) Ask the user for input file name.

(b) Load the data into a numpy array, call it grid.

(c) Set variables maxRow and maxCol to 0.

(d) Use a nested loop to consider every number in the grid, looping through rows in the outer loop and columns in the inner loop.

(e) If the current number (grid[currentRow, currentColumn]) is greater than the number at grid[maxRow, maxCol], set maxRow to the current row and set maxCol to the current column.

(f) Return maxRow and maxCol.

6. Fill in the Python code below for the function, `duplicates()`, that takes a list of names and returns a list with the names that occurred more than once.

Answer:

```
def duplicates(names):
    """
    Takes a list of names and returns a list of the duplicate names
    @param names:  a list of names
    @return: the names that occurred more than once
    """

    #Set up an empty dictionary, called new_dict:
    new_dict = {}
    #Set up an empty list, called dup_names:
    dup_names = []

    for name in names:
        #If name is in the dictionary:
        if name in new_dict:
            #Increment the count:
            new_dict[name] += 1
            #Check if it's occurred twice:
            if new_dict[name] == 2:
                #Add it to the duplicate name list:
                dup_names.append(name)
        else:
            #Create a new dictionary entry for name with value 1
            new_dict[name] = 1
    #Return the duplicate name list:
    return dup_names
```

7. Write a **complete Python program** that makes an interactive map using Plotly Express. Your program should ask the user for:

   - A list of place names,
   - A list of latitudes,
   - A list of longitudes, and
   - The name for the output (HTML) file.

   and save the resulting map, with the entered place names as the hover text at the locations specified saved to the output file.

   Hint: Build a DataFrame from the inputted lists and then use `px` *to create & save the map.*

   Answer:

```
#Mock Exam, S25, #7
import plotly.express as px
import pandas as pd
#Read in data:
name_str = input('Enter names, separated by spaces: ')
lat_str = input('Enter latitudes, separated by spaces: ')
lon_str = input('Enter longitudes, separated by spaces: ')
#Need to split up the inputted strings into lists:
names = name_str.split(' ')
lats = lat_str.split(' ')
lons = lon_str.split(' ')
#Set up a dictionary of the lists (used to make df):
data = {'latitude': lats, 'longitude': lons, 'name': names}
#Make a DataFrame of the dictionary:
df = pd.DataFrame(data)
#Use column names of df for keyword args:
fig = px.scatter_map(df,
                     lat="latitude",
                     lon="longitude",
                     hover_name="name")
#Save the output:
file_name = input('Enter output file name: ')
fig.write_html(file_name)
```

8. (a) Consider the following MIPS program:

```
ADDI $s0, $zero, 2
ADD $s1, $s0, $s0
SUB $s2, $s1, $s0
ADD $s3, $s1, $s2
```

   After the program runs, what is the value stored in:

$s1 register	$s2 register	$s3 register
Answer: 4	**Answer:** 2	**Answer:** 6

(b) Consider the MIPS code:

```
1  ADDI $sp, $sp, -5
2  ADDI $t0, $zero, 80
3  ADDI $s2, $zero, 88
4  SETUP: SB $t0, 0($sp)
5  ADDI $sp, $sp, 1
6  ADDI $t0, $t0, 2
7  BEQ $t0, $s2, DONE
8  J SETUP
9  DONE: ADDI $t0, $zero, 0
10 SB $t0, 0($sp)
11 ADDI $sp, $sp, -5
12 ADDI $v0, $zero, 4
13 ADDI $a0, $sp, 0
14 syscall
```

Answer:

i) How many letters are printed?	**4**
ii) What is the first letter printed?	**P**
iii) What is the whole message printed?	**PRTV**
iv) Detail the changes needed to the code to print the message in reverse:	**Line 2: Start t0 at 86.** **Line 3: Start s2 at 78.** **Line 6: Subtract 2 from t0.**

9. (a) What is the output:

```
#include <iostream>
using namespace std;
int main()
{
    for (int i = -5; i < 15; i += 5) {
       cout << i << endl;
    }
    return 0;
}
```

Answer:

```
-5
0
5
10
```

(b) Fill in the missing code to yield the output:

```
#include <iostream>
using namespace std;
int main()
{
    int n=12, m=-5;

    while(n+m [          ]){

        cout << n << " " << m << endl;
        n-=2;
        m++;
    }
    return 0;
}
```

Answer:

```
while(n+m > 0)
```

(c) What is the output:

```
#include <iostream>
using namespace std;
int main()
{
    int size = 5;
    for (int i = 1; i <= size; i += 2)
    {
        for (int j = 0; j < (size - i)/2; j++)
            cout << " ";
        for (int j = 0; j < i; j++)
            cout << "*";
        cout << endl;
    }
    return 0;
}
```

Answer:

```
   *
  ***
 *****
```

10. (a) Translate the Python into a **complete** C++ program:

Python program:

```
num = 0
while num <= 0:
    mess = int(input("Enter positive: "))
print("Your number:", num)
```

C++ program:

Answer:

```
#include <iostream>
using namespace std;
int main()
{
  int num =  0;
  while (num <= 0)
  {
    cout << "Enter positive:";
    cin >> num;
  }
  cout << "Your number: " << num;
  return 0;
}
```

(b) Write a C++ program that asks the user for the starting amount, and interest rate and prints out the yearly balance of a savings account, for 5 years.

A sample run:

```
Please enter the starting amount: 3000
Please enter the interest rate:  0.03
Year 1  3090.00
Year 2  3182.70
Year 3  3278.18
Year 4  3376.53
Year 5  3477.82
```

Answer:

```
#include <iostream>
using namespace std;

int main()
{
    float balance;
    cout<< "Enter starting amount: ";
    cin >> balance;

    for (int i = 1; i <= 5; i++)
    {
        balance = balance * 1.03;
        cout<< "Year " << i <<"\t" << total << "\n";
    }
}
```