Answer Key:

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

20 December 2021

1. (a) Fill in the code below to produce the Output on the right:

```
pioneers = "Lovelace,Ada-Fleming,Williamina-Hopper,Grace"
```

i.
```
names = [          ]
```

Answer Key:

```
pioneers.split('-')
```

ii.
```
for n in names:
    print(          )
```

Answer Key:

```
n.split(',')[0]
```

(b) Consider the following shell commands:

```
$ pwd
/usr/student
$ ls
classes.csv grades.csv  hello.py  hw60.py
```

i. What is the output for:

```
$ mkdir projects
$ mv *py projects
$ ls
```

Answer Key:

```
classes.csv grades.csv projects
```

ii. What is the output for:

```
$ cd projects
$ ls | grep hw
```

Answer Key:

```
hw60.py
```

iii. What is the output for:

```
$ cd ../
$ pwd
```

Answer Key:

```
/usr/student
```

2. (a) Select the color corresponding to the rgb values below:

Answer Key:

i. `rgb = (255, 0, 0)`
☐ black [✓ red] ☐ white ☐ gray ☐ purple

ii. `rgb = "#AB00AB"`
☐ black ☐ red ☐ white ☐ gray [✓ purple]

iii. `rgb = (0.5, 0.5, 0.5)`
☐ black ☐ red ☐ white [✓ gray] ☐ purple

iv. What is the 5-bit binary number equivalent of Decimal 24?

`Decimal 24 = Binary`

1	**1**	**0**	**0**	**0**

v. What is the Decimal number equivalent to Hexadecimal 1C?

`Hexadecmal 1C = Decimal`

2	**8**

(b) Given the list `symbols` below, fill in the code to produce the Output on the right:

```
symbols = ['*', '#', '+', '$', '%']
```

i. **Answer Key:**

```
for i in range( 2 ):
    for j in range( 4 ):
        print(symbols[j], end=" ")
```

Output:

```
* # + $ * # + $
```

ii. **Answer Key:**

```
for j in range( 4, -1, -2 ):
   print(j, end=" ")
```

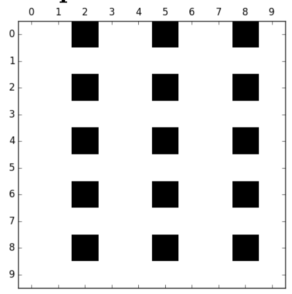
Output:

```
% + *
```

Answer Key:

iii.
```
import numpy as np
import matplotlib.pyplot as plt
im = np.ones( (10,10,3) )
im[0::2 , 2::3 , :]  = 0
plt.imshow(im)
plt.show()
```

Output:



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

i.
```
in1 = True
in2 = True
out = not( in1 and in2)
```

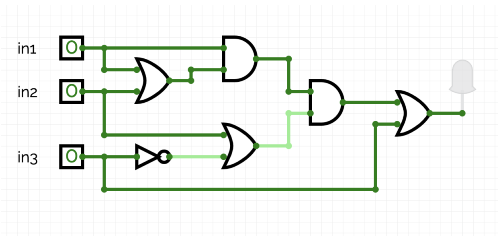
Answer Key:
```
out = False
```

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

Answer Key:
```
out = True
```

iii.
```
in1 = False
in2 = True
in3 = not in1 and in2
out = not in2 or not in3
```

Answer Key:
```
out = False
```

iv. 

```
in1 = True
in2 = False
in3 = False
```

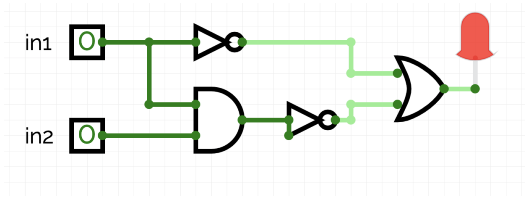
Answer Key:

```
out = True
```

(b) Draw a circuit that implements the logical expression:

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

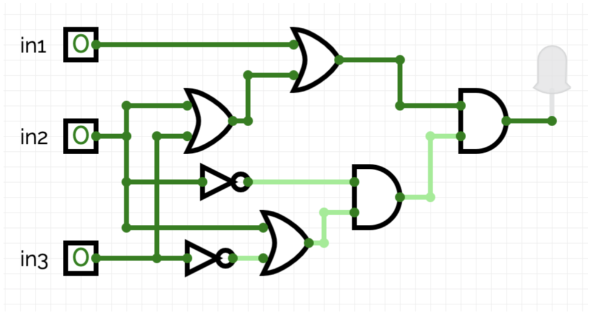
Answer Key:



(c) Fill in the circuit with the gate-symbol or gate-name that implements the logical expression:

```
(in1 or (in2 or in3)) and ((not in2) and (in2 or (not in3))
```

Answer Key:



4. Consider the following functions:

```
import numpy as np

def find_all(grid, n):
  for i in range(grid.shape[0]):
    for j in range(grid.shape[1]):
      if compare(grid[i,j], n):
        print(grid[i,j])
```

```
def compare(x, num):
   return x % num == 0

def main():
   table = np.array([[1, 2, 3, 4],
                     [15, 20, 25, 30],
                     [5, 10, 50, 75]])
   find_all(table, 10)
```

(a) What are the formal parameters for `compare()`?

Answer Key: x, num

(b) What are the actual parameters for `find_all()`?

Answer Key: table, 10

(c) How many calls are made to `compare()` after calling `main()`?

Answer Key: 12

(d) What is the output after calling `main()`?

i. **Output:**

Answer Key:

```
20
30
10
50
```

5. Design an algorithm that, given an image, outputs the number of pixels that are considered dark based on some user-provided threshold for darkness.
Libraries:

Answer Key: matplotlib.pyplot and numpy
Input:

Answer Key: The name of the image file and the threshold
Output:

Answer Key: The number of dark pixels **Design Pattern:**

Answer Key: ☐ Search ☐ Find Min ☐ Find Max **X** Find All **Principal Mechanisms (select all that apply):**

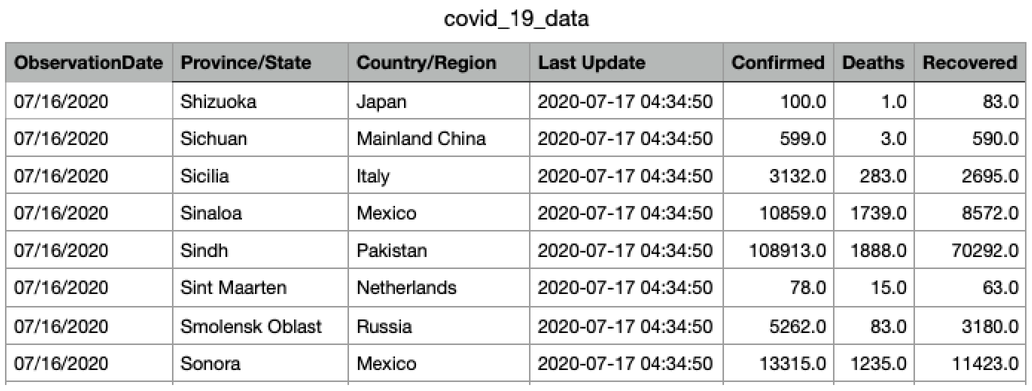
Answer Key: ☐ Search ☐ Single Loop **X** Nested Loop **X** Conditional (if/else) statement
XIndexing / Slicing ☐ `split()` ☐ `groupby()`

Process (as a concise and precise LIST OF STEPS / pseudocode):
(Assume libraries have already been imported.)

Answer Key:

(a) Ask the user for image file name
(b) Ask the user for darkness threshold
(c) Read the image in a numpy array, call it `img`
(d) Start a count of dark pixels at 0
(e) Use a nested loop to consider every pixel in `img` looping for rows in outer loop and columns in inner loop
    i. if the current pixel's red and green and blue channels are all less than or equal to the threshold, increment the count
(f) Return the count

6. Consider the `covid_19.csv` dataset that reports the number of observed COVID-19 cases in different countries by observation date. A snapshot given in the image below:



covid_19_data

ObservationDate	Province/State	Country/Region	Last Update	Confirmed	Deaths	Recovered
07/16/2020	Shizuoka	Japan	2020-07-17 04:34:50	100.0	1.0	83.0
07/16/2020	Sichuan	Mainland China	2020-07-17 04:34:50	599.0	3.0	590.0
07/16/2020	Sicilia	Italy	2020-07-17 04:34:50	3132.0	283.0	2695.0
07/16/2020	Sinaloa	Mexico	2020-07-17 04:34:50	10859.0	1739.0	8572.0
07/16/2020	Sindh	Pakistan	2020-07-17 04:34:50	108913.0	1888.0	70292.0
07/16/2020	Sint Maarten	Netherlands	2020-07-17 04:34:50	78.0	15.0	63.0
07/16/2020	Smolensk Oblast	Russia	2020-07-17 04:34:50	5262.0	83.0	3180.0
07/16/2020	Sonora	Mexico	2020-07-17 04:34:50	13315.0	1235.0	11423.0

Fill in the Python program below:

Answer Key:

```
#Plots number of recovered cases in Italy by observation date

#Import the libraries for data frames and plotting data
import pandas as pd
import matplotlib.pyplot as plt

#Prompt user for input file name:
csvFile = input('Enter CSV file name: ')

#Read input data into data frame:
```

```
df = pd.read_csv(csvFile)

#Groups the data by Country/Region to extract observations in Italy
italy = df.groupby('Country/Region').get_group('Italy')


#Plot the number of recovered cases over time (observation date)
italy.plot( x = 'ObservationDate', y = 'Recovered')
plt.show()
```

7. Write a **complete Python program** that prompts the user for the name of a .csv file and the names of latitude and longitude columns and generates an interactive .html map with markers found at each geographical location extrated from the .csv file.

   Answer Key:

```
#Import the packages for dataframes and for generating html maps
import pandas as pd
import folium

#Ask user for the name of csv file and store in variable in_file
in_file = input("Enter the name of csv file: ")

#Ask user for the name of latitude and longitude columns
#and store in variables lat and long respectively
lat = input("Enter the name of latitude column: ")
long = input("Enter the name of longitude column: ")

#Read the csv file into a dataframe and store it in variable df
df = pd.read_csv(in_file)

#Create a map and store in variable map
map = folium.Map()

#Loop through all the rows in the dataframe, create a marker with
#values found in columns lat and long, add marker to the map
for index,row in df.iterrows():
  lt = row[lat]
  lg = row[long]
  mark = folium.Marker([lt, lg])
  mark.add_to(map)

  #Save the map to file named map.html
  map.save(outfile='map.html')
```

8. (a) What does the MIPS program below print:

   Answer Key:

   ```
   abcdefg
   ```

   (b) Modify the program to print out 15 consecutive letters in decreasing order ('Z' down to 'L'). Shade in the box for each line that needs to be changed and rewrite the instruction below.

   Answer Key:

```
# Print the alphabet
ADDI $sp, $sp, -16 # Set up stack
ADDI $s3, $zero, 1 # Store 1 in a register
ADDI $t0, $zero, 90 # Set $t0 at 90 (Z)
ADDI $s2, $zero, 15 # Use to test when you reach 15
SETUP: SB $t0, 0($sp) # Next letter in $t0
ADDI $sp, $sp, 1 # Increment the stack
SUB $s2, $s2, $s3 # Decrement the counter by 1
SUB $t0, $t0, $s3 # Decrement the letter by 1
BEQ $s2, $zero, DONE # Jump to DONE if s2 == 0
J SETUP         # Else, jump back to SETUP
DONE: ADDI $t0, $zero, 0 # Null (0) to terminate string
SB $t0, 0($sp) # Add null to stack
ADDI $sp, $sp, -15 # Set up stack to print

ADDI $v0, $zero, 4 # 4 is for print string
ADDI $a0, $sp, 0 # Set $a0 to stack pointer
syscall # Print to the log
```

9. Fill in the C++ programs below to produce the Output on the right.

```
#include <iostream>
using namespace std;
int main()
{

    int num = [        ];
```

Answer Key:

```
5
```

(a)
```
    for(int i = 0; i <=30; [        ]){
```

Answer Key:

```
i += num
```

```
       num += 5;
       cout << i << " " << num << endl;
    }
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    double num = 0;
    double tot = 0;
```

(b)
```
    while ( [        ]){
        cout <<"Please enter amount\n";
        cin >> num;
        tot += num;
    }
    cout <<"The total is " <<  tot << endl;
    return 0;
}
```

Answer Key:

```
tot <= 10
or
tot < 11
```

```
#include <iostream>
using namespace std;
int main(){

    for (int i = 1;              i++){
```

Answer Key:

```
  i < 5;
  or
  i <= 4;
```

(c)

```
        for (int j = 0;              j++ ){
```

Answer Key:

```
  j < i;
```

```
            if(j % 2 == 0)
                cout << "X";
            else
                cout << "O";
        }
        cout << endl;
    }
    return 0;
}
```

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

```
#Python Loops
for i in range(0,101,25):
    print(i+5, i-5)
```

Answer Key:

```
 //C++ Loops
#include <iostream>
using namespace std;
int main(){
    for(int i = 0; i < 101; i +=25){
        cout << i+5 << " " << i-5 << endl;
    }
    return 0;
}
```

(b) Parsec is a unit of distance used in astronomy, equal to 3.26 light years and 30.9 trillion kilometers. One parsec corresponds to the distance at which the mean radius of the earth's

orbit subtends an angle of one second of arc.
Write a **complete C++ program** that asks the user for the number of parsecs and prints the corresponding number of light years and kilometers.

Answer Key:

```
//include library and namespace
#include <iostream>
using namespace std;

//main function signature
int main(){

    //initialize variables
    float parsec = 0.0;
    float light_years = 0.0;
    float kilometers = 0.0;

    //obtain input
    cout << "Enter the number of parsecs: ";
    cin >> parsec;

    //calculate light years
    light_years = parsec * 3.26;

    //calculate kilometers
    kilometers = parsec * 30900000000000;

    //output conversions
    cout << parsec << " parsecs = " << light_years << " lgiht years.\n";
    cout << parsec << " parsecs = " << kilometers << " kilometers.\n";

    return 0;
}
```