Computer Programming Final Exam
School of Information Technology & Engineering, AAiT **Time Allotted: 2hrs**
Full Name ____ ID _____________ Campus _________ Section ____

Guidline for marking the answer papers:

1. **Question 1 - 8 ask for output of a code. As such, if the student writes the exact output, he/she will get full point (3 points). Otherwise, it will be just no points at all.**
2. **Question 9 - 13 ask for solving a problem through writing code. As you know, there are many ways of writing a particular objective. I just put some just to show a solution and the students might have different implementations. To achieve uniformity along all sections, please follow the following ways to mark the answers:**
   a. **If the code generally follows correct python coding structure, give it 1 point (even if the code doesn't work)**
   b. **If the code generally has the sense of close to a solving the code, give it 1 to 2 points. (If the code is correctly write, given it 3 points). 'Sense of closeness to solving the code' is vague. But this is where you need your judgement.**
   c. **If the code works and solves the problem, give it a full point. Not that there might be minor, ignoreable mistakes (Use your own judjemnt here too).**

The general aim is, as more and more of the written code is close to solving the problem, the student should get close to the full point. With that mindset, please be considerate and mark the answers.

1. What will be the value of a Python expression? Show the steps: `5 * 3 ** 2 > 10 and 4 + 6 == 11` ***(3 pts)***

   Ans: False

2. What is the difference between a keyword and an identifier? ***(3 pts)***

   Ans: A keyword is a reserved word in a programming language that has a special meaning to the compiler. Keywords cannot be used as identifiers as identifier is a name given to a variable, function, class, or other programming entity. Identifiers must be unique within the scope in which they are declared.

3. What is the expected output when the following code snippet is executed with an input value of 8? ***(3 pts)***

```
n = int(input("Enter an integer (n): "))
for i in range(1, n + 1, 2):
    print(i, i + 1)
```

   Ans:

```
1 2
3 4
5 6
7 8
```

4. What is the output of the following for loop and range() function ***(3 pts)***

```
for num in range(-2,-5,-1):
      print(num, end=", ")
```

Ans:
-2, -3, -4,

5. What will be the output of the code? ***(3 pts)***

```
def mystery_function(data):
      result = data[0]
      for item in data[1:]:
            result = result + item
            return result

input_list = [1, 2, 3.0, 4, 5]
output = mystery_function(input_list)
print(type(output), output)
```

```
Ans:
<class 'int'> 3
```

6. Write the output of the following code: ***(3 pts)***

```
lst = [1, 2, 3]
lst.append(lst)
print(len(lst))
print(lst[3][2])
```

```
Ans:
4
3
```

7. What is the output? ***(3 pts)***

```
for string in "AAU Student":
      if string == "s" or string == "u" or string == "t":
            continue
      print ('Current Letter:', string)
```

```
Ans:
Current Letter: A
Current Letter: A
Current Letter: U
Current Letter:
Current Letter: S
Current Letter: d
Current Letter: e
Current Letter: n
```

8. Write the output of the following code. ***(3 pts)***

```
tuple1=(11,22,33,44,55,66)
list1=list(tuple1)
new_list=[]
```

```
for i in list1:
    if i%2==0:
        new_list.append(i)
new_tuple=tuple(new_list)
print(new_tuple)
```

Ans:
(22, 44, 66)

9. Write a Python function called **check_sign** that takes three integer values as parameters: **a, b,** and **c**, and a boolean parameter **strict**. Method signature: **`check_sign(a, b, c, strict)`** ***(4 pts)***

The function should return **True** under the following conditions:
If **strict** is ***True***, return ***True*** if all three values have the same sign (either all positive or all negative).
If **strict** is ***False***, return ***True*** if at least two values have the same sign (either both positive or both negative).

```
# Examples:
check_sign(1, -2, 3, True) --> False
check_sign(-1, -2, -3, False) --> True
```

`Ans:`

```python
def check_sign(a, b, c, strict):
    positive_count = 0
    negative_count = 0
    if a > 0:
        positive_count += 1
    elif a < 0:
        negative_count += 1

    if b > 0:
        positive_count += 1
    elif b < 0:
        negative_count += 1

    if c > 0:
        positive_count += 1
    elif c < 0:
        negative_count += 1

    if strict:
        return positive_count == 3 or negative_count == 3
    else:
        return positive_count >= 2 or negative_count >= 2

# Examples
print(check_sign(1, -2, 3, True) ) # False
print(check_sign(-1, -2, -3, False) ) # True
```

10. Write a Python function **remove_duplicates(input_list)** that takes a list **input_list** as input and removes any duplicate elements, preserving the original order. The function should return a new list with duplicates removed.
remove_duplicates([1, 2, 2, 3, 4, 4, 5]) # Should return [1, 2, 3, 4, 5] ***(5 pts)***

```
def remove_duplicates(input_list):
    unique_elements = []
    seen = set()
    for item in input_list:
        if item not in seen:
            unique_elements.append(item)
            seen.add(item)
    return unique_elements

# Example usage:
result = remove_duplicates([1, 2, 2, 3, 4, 4, 5])
print(result) # Output: [1, 2, 3, 4, 5]
```

11. Write a program that sorts a list of names based on their length. ***(5 pts)***

Output = sort_names_by_length(["Alice", "Bob", "Carol", "Dave", "Eve"])
print(output) # ['Bob', 'Eve', 'Dave', 'Alice', 'Carol']

```
def sort_names_by_length(names):
    n = len(names)
    for i in range(n):
        for j in range(0, n-i-1):
            if len(names[j]) > len(names[j+1]):
                names[j], names[j+1] = names[j+1], names[j]
    return names

# Example usage:
output = sort_names_by_length(["Alice", "Bob", "Carol", "Dave", "Eve"])
print(output) # Output: ['Bob', 'Eve', 'Dave', 'Alice', 'Carol']
```

12. Write a Python function that finds the largest prime factor of a given positive integer number [CASTING IS NOT ALLOWED]. ***(6 pts)***

```
def largest_prime_factor(n):
    largest_prime = -1
    i = 2
    while i * i <= n:
        While n % i == 0:
            largest_prime = i
            n = n // i
        i = i + 1
    if n > 1:
        largest_prime = n
    return largest_prime
n = 6
print(largest_prime_factor(n))
# 3
```

13. Write a Python function **count_words** to count the number of occurrences of each word in a string after removing punctuations found in the **punc** variable. The output should be a dictionary containing the frequency of each word. ***(6 pts)***

```
punc = ['!', '"', '#', '$']
str = "the apple,orange,banana and the grapes in the fridge."

output = count_words(str, puncs)
print(output)

{'apple':1,  'orange': 1, 'banana': 1, 'and': 1, 'grapes: 1, 'in': 1, 'the': 3,
'fridge.': 1}
```

```
def count_words(input_str, punc):
    for char in punc:
        input_str = input_str.replace(char, '')
    word_list = input_str.lower().split()
    word_count = {}
    for word in word_list:
        if word in word_count:
            word_count[word] += 1
        else:
            word_count[word] = 1
    return word_count

# Example usage:
punc = ['!', '"', '#', '$']
input_str = "the apple,orange,banana and the grapes in the fridge."
output = count_words(input_str, punc)
print(output)
```