2024 EXAMINATIONS

Lancaster University

Part I

COMPUTING AND COMMUNICATIONS

SCC.111 – Software Development – In-Person, Online [1.5hrs]

Candidates are asked to answer THREE questions from THREE; each question is worth a total of 25 marks.

Question 1

1.a.

i. Which of the following is NOT a valid variable name in C?

a. 111_scc
b. scc_111
c. _scc111
d. SCC111

[1 Mark]

ii. Which of the following declarations **follows** the scc.111 variable naming convention?

a. LANCASTERUNI
b. lancasterUni
c. LancasterUni
d. lancasteruni

[1 Mark]

iii. What does the **sizeof** operator return in C?

a. The size of a variable in bytes
b. The type of a variable
c. The address of a variable
d. The value of a variable

[1 Mark]

iv. Which of the following is **NOT** a valid way to declare and initialise an array in C?

a. int arr[5];
b. int arr[] = {1, 2, 3, 4, 5};
c. int arr[5] = {1, 2, 3, 4, 5};
d. int arr[5] = {0};
e. arr = {1, 2, 3, 4, 5};

[1 Mark]

v. How do you declare a pointer to an integer variable in C?

a. int p;
b. integer *p;
c. int *p;
d. p int;

[1 Mark]

Question 1 continues on next page…

Question 1 continued.

vi. Given an integer variable x (int x = 111), what is the result of the following printf statement (printf("%d\n", x);)

a. The string "%d"
b. 111
c. %d\n
d. The address of variable x

[1 Mark]

vii. What is the purpose of the **break** statement in C?

a. To exit a loop or switch statement
b. To jump to a specific break handler
c. To exit the program
d. To skip the current iteration of a loop

[2 Marks]

1.b. Given the following code, identify the correct sequence of output to the terminal:

```
1   int x = 0;
2
3   void print(int x)
4   {
5       printf("x = %d\n", x);
6   }
7
8   int main()
9   {
10      int x = 1;
11
12      print(2);
13      print(x);
14      printf("x = %d\n", x);
15  }
```

Question 1 continues on next page…

Question 1 continued.

a. x = 0
   x = 0
   x = 0

b. x = 2
   x = 1
   x = 1

c. x = 2
   x = 1
   x = 0

d. x = 2
   x = 0
   x = 0

[2 Marks]

1.c. Given the following C code, 'dry run the code' on paper and analyse the behaviour of the algorithm if 'func' is called with the literal parameter 8.

```
1   int func(int positiveInt)
2   {
3       int l = 0, m, h = positiveInt + 1;
4
5       while (l != h - 1)
6       {
7           m = (l + h) / 2;
8
9           if (m * m <= positiveInt)
10              l = m;
11          else
12              h = m;
13      }
14
15      return l;
16  }
```

i. What is the value of 'l' that is returned?

a. 2
b. 3
c. 4
d. 0
e. 8

[1 Marks]

Question 1 continues on next page…

Question 1 continued.

ii. How many iterations of the loop does it take to find the answer?

a. 2
b. 8
c. 0
d. 4
e. 3

[1 Mark]

iii. Based on your understanding of what the algorithm does, which of the following statements is true?

a. The answer returned is the closest value to the correct answer
b. The efficiency of the algorithm is all that matters
c. The algorithm is only suitable for use where the result should be a whole number
d. The answer is a good approximation to the correct answer

[1 Mark]

1.d. A sloppy programmer has hurriedly written the following code designed to find the largest of a sequence of numbers. They've not tested their code properly! Identify the line numbers containing the errors. Note negative marks are awarded for incorrectly identified lines.

```
1   #include <stdio.h>
2   #include <stdlib.h>
3
4   #define SERIES_LENGTH 3
5
6   int max(int x, int y)
7   {
8       return x <= y ? x : y;
9   }
10
11  int main()
12  {
13      int numberSeries[] = { 30, 20, 10 };
14
15      int largest;
16
17      for (int index = 1; index <= SERIES_LENGTH; index++)
18          largest = max(largest, numberSeries[index]);
19
20      printf("Largest = %c\n", largest);
21
22      return;
23  }
```

[5 Marks]

Question 1 continues on next page…

Question 1 continued.

1.e. You are working on a text editor code base with your team. You've just hired a new junior C developer who's a bit rusty at C. To help onboard them, you have been asked to help them write a short C program (including a main function) to show them how to implement the 'count words' functionality for the editor.

The program should take words separated by spaces as input from stdin. Your program should print a single line of output 'Words = ' and the number of words. See the following short illustrative dialogue:

$ ./wordcount
hello world
Words = 2

You can assume stdin.h and ctype.h are already #included, so you have fgetc, printf and isspace available to you. Recall that fgetc(stdin) either returns a character code or EOF.

For example:

Result
Words = 2

```c
#include <stdio.h>
#include <ctype.h>

int main()
{
    int words = 0;

    // Your code here

    printf("Words = %d\n", words);

    return 0;
}
```

[8 Marks]

[Total 25 Marks]

Question 2

2.a. Consider the following statements about Object Oriented programming concepts carefully. For each statement, complete the sentence correctly by using the drag and drop markers below.

Note:

- You may use the same marker twice if you wish.
- Marks will be provided for each correct answer.
- Incorrect answers may incur a penalty.
- You need not need to answer all questions to proceed.

1. Encapsulation aims to [ ] code from outside a class from directly [ ] a(n) [ ] in that class. This improves code [ ].
2. Inheritance allows a class to [ ] another. Each [ ] and [ ] from the [ ] becomes part of the new class.
3. Polymorphism states that a [ ] of another [ ] can be treated as a class of the same [ ].
4. Method overriding means that a [ ] defined in a [ ] will replace one defined in the [ ] with the same [ ].
5. Method overloading occurs when we have more than one [ ] with the same [ ] in the same [ ].
6. Recursion is defined as a [ ] that can [ ] [ ].
7. Languages like Java, C# and Python have more [ ] than C/C++ because they do not have the concept of a [ ].
8. Generics allow the creation of classes that take a [ ] parameter. This improves the [ ] of the class.

attribute | child | class | method | name | pointer

signature | subclass | superclass | type

another | itself | one

performance | reusability | safety | simplicity

call | extend | package | reduce | replace

adding | changing | removing

encourage | prevent | require

[6 Marks]

Question 2 continues on next page…

Question 2 continued.

2.b. Consider the following small C++ program.

Cat.cpp

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <string.h>
4
5   class Cat{
6       private:
7           char* furColour;
8           char* name;
9       public:
10          Cat(char* colour, char* n);
11          Cat(Cat &c, Cat &c2);
12          void show();
13  };
14
15  Cat::Cat(char* colour, char* n){
16      furColour = colour;
17      name = n;
18  }
19
20  Cat::Cat(Cat &c, Cat &c2){
21      char* n = (char *)malloc(100);
22      strcpy(n, c.name);
23      strcat(n, c2.name);
24      name = n;
25      furColour = c.furColour;
26  }
27
28  void Cat::show(){
29      printf("My name is %s, and I am %s\n", name, furColour);
30  }
31
32
33  int main(){
34      Cat a ((char *)"Grey",(char *)"Luna");
35      Cat b ((char *)"Yellow",(char *)"Sol");
36      Cat c(a, b);
37
38      c.show();
39  }
```

For each line listed below, select the item from the pull down list with the most accurate description.

Question 2 continues on next page…

Question 2 continued.

Lines :
5
7
8
10
11
12
15
20
28
34
35
36
38

Choose from the following answers.

Attribute
Method Definition
Method Invocation
Constructor Declaration
Method Declaration
Class Definition
Object Instance
Constructor Definition

[2 Marks]

Question 2 continues on next page…

2.c. Consider again this program. It is identical to the previous question.

Cat.cpp

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

class Cat{
    private:
        char* furColour;
        char* name;
    public:
        Cat(char* colour, char* n);
        Cat(Cat &c, Cat &c2);
        void show();
};

Cat::Cat(char* colour, char* n){
    furColour = colour;
    name = n;
}

Cat::Cat(Cat &c, Cat &c2){
    char* n = (char *)malloc(100);
    strcpy(n, c.name);
    strcat(n, c2.name);
    name = n;
    furColour = c.furColour;
}

void Cat::show(){
    printf("My name is %s, and I am %s\n", name, furColour);
}


int main(){
    Cat a ((char *)"Grey",(char *)"Luna");
    Cat b ((char *)"Yellow",(char *)"Sol");
    Cat c(a, b);

    c.show();
}
```

What would this program print to the screen when executed? BE PRECISE. You must match the answer perfectly, so check your answer before submitting.

[2 Marks]

Question 2 continues on next page…

Question 2 continues.

2.d. Consider again this program. It is identical to the previous question.

Cat.cpp

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

class Cat{
    private:
        char* furColour;
        char* name;
    public:
        Cat(char* colour, char* n);
        Cat(Cat &c, Cat &c2);
        void show();
};

Cat::Cat(char* colour, char* n){
    furColour = colour;
    name = n;
}

Cat::Cat(Cat &c, Cat &c2){
    char* n = (char *)malloc(100);
    strcpy(n, c.name);
    strcat(n, c2.name);
    name = n;
    furColour = c.furColour;
}

void Cat::show(){
    printf("My name is %s, and I am %s\n", name, furColour);
}


int main(){
    Cat a ((char *)"Grey",(char *)"Luna");
    Cat b ((char *)"Yellow",(char *)"Sol");
    Cat c(a, b);

    c.show();
}
```

Are there any problems in this program? If you think not, just answer "NO". If you think there is a problem, answer "YES" followed by a single sentence describing the issue.

[2 Marks]

Question 2 continues on next page…

Question 2 continued.

2.e. Imagine you have been asked to create a simple Java program to record the results of the annual Roses sporting competition between Lancaster University and the University of York.

Write a class called University. This class should contain attributes to record:

name : The name of the University
points : The number of points the University has scored in total (you may assume the points are all whole numbers).
Write a constructor for your University class, that initializes the name of the university to a given parameter, and initializes the number of points scored to zero.

Create accessor (get) methods for your two attributes, called getName and getPoints respectively.

Be sure to encapsulate your class correctly.

[5 Marks]

2.f. Copy the class you have written above into the box below (remember you can use ctrl-c and ctrl-v to copy/paste respectively). Then answer the following questions.

Write a second class called **SportsEvent**. This class should contain attributes to record:

- **name** **:** The name of the event.
- **pointsAvailable** **:** A whole number, representing the number of points the winner of this event will claim.
- **winner** **:** An object reference to a University object, indicating which University won this event.

Write a constructor for your **SportsEvent** class, that initializes the **name** and **pointsAvailable** attributes to values provided in parameters to the constructor **in that order.** The winner attribute should be initialized to null.

Create **accessor** (get) methods for the **pointsAvailable** and **winner** attributes respectively. Follow standard Java naming conventions.

Create a **mutator** (set) method for the winner attribute. Follow standard Java naming conventions.

Be sure to **encapsulate** your class correctly.

[4 Marks]

Question 2 continues on next page…

2.g. Copy all of the classes you have written above into the box below (remember you can use ctrl-c and ctrl-v to copy/paste respectively). Then answer the following questions.

Write **two** additional methods for your **University** class, that meet the following specification:

- **calculatePoints :** This method should take an array of **SportsEvent** objects as its only parameter and return nothing. It should update the University's **points** attribute to be the total of all the **SportsEvent points** won by that university, as defined by the items in the array.
- **beat :** This method should take another **University** object as its only parameter and should return **true** if this University has **more** points than the University given as a parameter, and **false** otherwise.

[4 Marks]

[Total 25 Marks]

Question 3

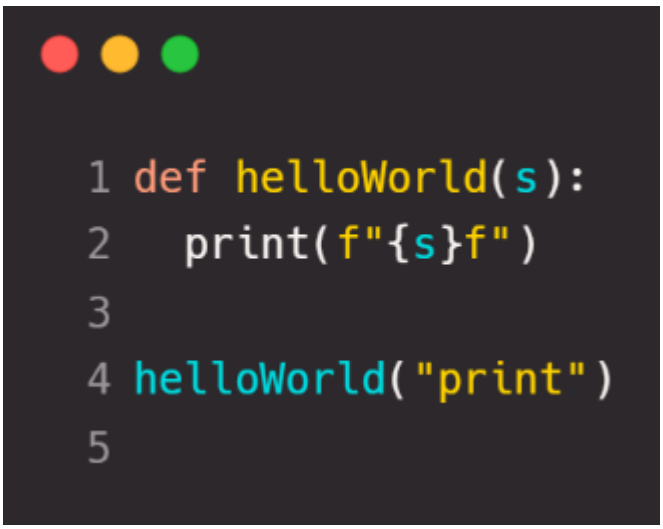
3.a.

i. Which of the following is NOT a built-in data type in Python?

a. Dictionary
b. Array
c. List
d. NoneType
e. String

[1 Mark]

ii. What is the output of the following code snippet?



```
1 def helloWorld(s):
2   print(f"{s}f")
3
4 helloWorld("print")
5
```

a. NameError
b. TypeError
c. fprintf
d. helloWorld
e. SyntaxError
f. print f
g. print
h. printf

[1 Mark]

iii. Which of the following is a built-in function in Python to find the length of a list?

a. len()
b. count()
c. number_of()
d. size()
e. length()

[1 Mark]

Question 3 continues on next page…

Question 3 continued.

iv. How many bytes do integers occupy in memory in Python?

a. 1 byte
b. 8 bytes
c. Depends on the integer size
d. 2 bytes
e. 4 bytes

[1 mark]

vi. What is the purpose of the __init__ method in a class in Python? **[1 mark]**

a. None of the given options
b. Create a new object of the class
c. Define instance methods
d. Initialize class variables
e. Perform cleanup operations

[1 Mark]

vii. What is the output of the following code snippet?

a. Hello
b. Hel
c. ello
d. HelloHello
e. llo

[1 Mark]

vii. What does the super() function do in Python when used in a class? Check all boxes that apply. Negative marks apply for wrong answers. The minimum mark is 0. **[2 marks]**

a. Creates a superclass instance
b. Returns the instance of the parent class
c. Initializes the child class
d. Calls the parent class constructor
e. Calls a method in the child class
f. Calls the base class destructor

[2 Marks]

Question 3 continues on next page…

Question 3 continued.

3.b. Consider the following Python code snippet:

```
def process_data(data):
    result = [x ** 2 for x in data if x % 2 == 0 else x / 2]
    return result

```

This function **[Blank 1]** every **[Blank 2]** element in the input list data and **[Blank 3]** the **[Blank 4]** elements.

Choose from the following answers.

Divides by 2
Increments
Doubles
Squares
Odd
Negative
Even
Positive

[3 Marks]

3.c.

i. Complete the following program that calculates someone's age by subtracting their birth year from the current year and prints the calculated age to the console.

```
from datetime import *
testYear = 1969
def calculateAge (BLANK):
  today = datetime.now().date()
  (BLANK) = today.year
  age = currentYear - birthYear
  return (BLANK)

result = (BLANK)(testYear)
print('Your age is {} years old.'.format(BLANK))
```

[2 Marks]

Question 3 continues on next page…

Question 3 continued.

ii. Complete the following object-oriented Python code for a simple spaceship

```
(BLANK) Spaceship:
    # constructor

    def  (BLANK) (self, name, lasers):
        self.name = name
        self.lasers = lasers
    # this function shoot lasers at an astroid, as long as the spaceship has lasers.
    def shoot_laser(self, astroid):
        if (BLANK).lasers > 0:
            print(f"{self.name} fired a laser at {astroid.name}!")
            (BLANK).lasers -= 1
            astroid.destroy()
        (BLANK):
            print(f"{self.name} is out of lasers!")

    def __str__(self):
        return f"Spaceship(name={self.name}, lasers={self.lasers})"


asteria = Astroid("Asteria")

falcon = Spaceship("Falcon", 10)

(BLANK).shoot_laser(asteria)
```

[2 Marks]

Question 3 continues on next page…

3.d. Punch Club 1 🥊

In the last two questions, you will be working with a simple boxing game simulation. Each player has three statistics: strength, agility, and stamina. Your task is to implement functions that determine the damage dealt in an attack and simulate a fight between two players using these statistics.

Part 1: Calculate Hit Damage
Write a Python function calculate_hit_damage that determines the damage dealt by an attacker to a defender. The damage calculation should follow these rules:

The base damage is equal to the strength of the attacker.
If the attacker's agility is greater than or equal to the defender's agility, the damage is increased by 20%.
If the attacker's agility is less than the defender's agility, the damage is decreased by 10%.
Additionally, if the attacker's agility is more than twice the defender's agility, the calculated damage is then doubled.
But if the defender's agility is more than twice the attacker's agility, the attack is dodged, resulting in only 10% of the base damage.
Function Signature:

def calculate_hit_damage(attacker, defender):

Parameters:

attacker (dict): Dictionary representing the attacking player with keys 'strength', 'agility', and 'stamina'.
defender (dict): Dictionary representing the defending player with keys 'strength', 'agility', and 'stamina'.
Returns:
(float): The calculated damage.

Note 1: The tests are random generated numbers, do not hard code them as new ones will be generated during grading.
Note 2: You are required to implement the calculate_hit_damage function. Do not write code outside the function and do not implement other functions. You can only use built-in functions, so do not import packages.

[4 Marks]

Question 3 continues on next page…
Question 3 continued.

3.e. Punch Club 2 🥊

Even though the second part relies on the calculate_hit_damage function from the first part, your previous implementation is not used here. Therefore, any mistakes in your answer to Part 1 will not affect your score for Part 2.

Part 2: Simulate a Fight
Using the calculate_hit_damage function from Part 1, write a Python function simulate_fight that simulates a turn-based fight between two players. The fight simulation should follow these rules:

Both players start with health equal to 10 times their stamina.
Players take turns attacking each other. Player 1 attacks first.
The attack reduces the defender's health by the damage calculated from calculate_hit_damage.
The fight continues until the health of one of the players drops to 0 or below.
Function Signature:

def simulate_fight(player1, player2):
Parameters:

player1 (dict): Dictionary representing the first player with keys 'strength', 'agility', and 'stamina'.
player2 (dict): Dictionary representing the second player with keys 'strength', 'agility', and 'stamina'.
Returns:

(int): The number of the winning player (1 or 2).
Note: The tests are random generated numbers, do not hard code them as new ones will be generated during grading.

[6 Marks]

[Total 25 Marks]

---End of Paper---