Name: ______________________ **Student #:** ________________ **Section #:** ______

University of Saskatchewan
Department of Computer Science
CMPT 141.3
Midterm Examination
October 18, 2017

Marks: 66 **Time: 90 minutes**

Instructions:

- Don't panic!
- Write your name, student number, and section number on this page, and your student number on the top of *every* page.
- This exam has multiple choice questions which you will answer using OpScan sheets.
  - Write your name and your identifying information on the OpScan sheet in ink.
  - Use a pencil to indicate your answers.
  - When you finish this exam, hand the OpScan sheet **and this exam paper** in to the invigilator.
- This exam has written questions which you will answer in this exam booklet.
  - Write your answers in the spaces provided.
  - Don't make us hunt for your answer; draw a box or a circle around it if it is not 100% clear.
  - The last page in this exam is blank; you may use it for rough work.
  - Written questions that are not completed in pen cannot have their marking disputed.
- The mark value of each question is provided in the left margin.
- No aids of any kind are allowed for this exam.
  - International students may consult a dictionary. Please inform the invigilators if you wish to use a dictionary. Dictionaries are subject to inspection by invigilators.
- Read every question carefully!

Academic Honesty

This exam is an individual undertaking – cheating on an exam is considered a serious offence by the University and can be met with disciplinary action, including suspension or expulsion. By handing in this exam and supplementary op-scan sheet you affirm that this work is entirely your own.

It will be considered an academic offence if you take this examination paper from the exam room.

Page	1–6	7	8	9	10	11	12	**Total**
Marks								
Max	28	6	9	8	6	5	4	66

Part I — Multiple Choice

Choose the best answer for each question. Choose only one answer per question.

Section 1: Multiple Choice Questions

(1) 1. Which one of the following statements about algorithms is true?

A. An algorithm is a set of instructions that only computers can perform.

B. An algorithm with more than 4 actions has to be refined.

C. An algorithm can have conditionals or repetition, but not both.

D. An algorithm is always written in a programming language like Python.

E. An algorithm is a sequence of actions that describe how to perform a task or solve a problem.

(1) 2. Which one of the following statements about algorithms is **FALSE**?

A. Algorithms written in more-or-less regular English are called pseudocode.

B. Algorithms written in a programming language are called computer programs.

C. Algorithms written in pseudocode cannot be understood and carried out by a computer.

D. Algorithms written in pseudocode obey strict syntax so that a computer can carry them out.

E. Algorithms written as a computer program may be used for communication between human beings.

(1) 3. Which one of the following statements about abstraction and refinement is true?

A. Abstraction is bad for computers and good for people.

B. Refinement is the process of fixing errors in an abstract algorithm.

C. Refinement is something a computer can easily do.

D. Abstraction and refinement are essentially the same thing.

E. Abstraction allows computers to think about actions without worrying about how they are performed.

(1) 4. Which of the following Python expressions is an **atomic literal expression**? If none are atomic literal expressions, choose "None of the above."

A. `True`

B. `atomic == True`

C. `"true"`

D. `true`

E. None of the above.

(1) 5. Compound data is:

A. The smallest unit of data.

B. Data that can be subdivided.

C. Always arranged as a list.

D. A single data value.

E. Data which is very large.

(1) 6. Which of the following Python expressions has a compound **data value**? If none are compound, choose "None of the above."

A. `321`

B. `3 + 2 + 1`

C. `3.0 * 1.0`

D. `(3, 2, 1)`

E. None of the above.

(1) 7. Which of the following is **NOT** a data type?

A. Floating point

B. Dictionary

C. Boolean

D. Tuple

E. All of the above are data types

(1) 8. Which one of the following statements is true?

A. The expression `"12"` is an integer literal.

B. The expression `twelve` is a string literal.

C. The expression `12e-2` is a floating point literal.

D. The expression `12` is a floating point literal.

E. The expression `-12.0` is an integer literal.

(1) 9. Which of the following is **NOT** a valid Python expression?

A. `"Back " + 2 + " Back”`

B. `True and False`

C. `"Woof!" * 20`

D. `0 % 9`

E. `14e4 - 2`

(1) 10. Assume the Boolean variables `fire` and `water` describe the type of a Pokemon. Which of the following Python expressions means "the Pokemon is either fire type or water type, but not both"?

A. `fire or water`

B. `not (fire and water) and (fire or water)`

C. `not fire or not water`

D. `(not fire and not water) and (fire or water)`

E. None of the above

(1) 11. Which one of the following is a **valid** variable name (identifier)?

A. `three little pigs`

B. `3_little_pigs`

C. `"three_little_pigs"`

D. `three!little!pigs`

E. `_3_little_pigs`

(1) 12. Which one of the following statements about variable scope is **FALSE**?

A. Variables created inside a function only exist inside that function.

B. Two different functions can use the same variable names without interfering with each other.

C. Variables created outside any function are accessible from all parts of the program.

D. There is no way to access a variable created outside a function from within a function.

Assume that the following variable declarations and initializations are given for questions 14 — 20.

```
i1 = 10
i2 = 2
i3 = 0
f1 = 1.5
f2 = 4
f3 = 0.0
```

(1) 13. What is the value of the expression `24 / i2 + 10 * i2`?

A. 32.0 B. 4 C. 44 D. 4.0 E. 32

(1) 14. What is the value of the expression `24 // i2 + 10 * i2`?

A. 32.0 B. 4 C. 44 D. 4.0 E. 32

(1) 15. What is the value of the expression `i1 % f2 + f1`?

A. 2.5 B. 3.0 C. 3.5 D. 4.0 E. 4.5

(1) 16. What is the value of the variable `i2` after the following assignments statements in the order given:

```
f3 = i2
f1 = f2
f2 = f3
```

A. 0.0 B. 4 C. 1.5 D. 2 E. None

(1) 17. What is the value of the expression (use the original values, ignoring the effects of the previous question): `(0 == i1) or (i1 > 14)`?

A. `True` B. `False`

(1) 18. What is the value of the expression: `(10 != i1) and (i1 < 14)`?

A. `True` B. `False`

(1) 19. What is the value of the expression:
`not (i1 <= i1) and (i2 < f2) or (i1 + i2) > i2`?

A. `True` B. `False`

(1) 20. What does the term 'immutable' mean? What is an example of an 'immutable' data type?

A. Immutable means changeable. Booleans are immutable.

B. Immutable means changeable. Lists are immutable.

C. Immutable means not changeable. Tuples are immutable.

D. Immutable means not changeable. Dictionaries are immutable.

E. Immutable means hating mutants (like the X-Men). Strings are immutable.

(1) 21. All strings have a method named `upper`, that takes no arguments, and returns a new string with all lower case characters converted to upper case. Suppose we create a variable as follows:

```
diss = 'Someone move this walking carpet!'
```

Which of the following is a correct call to the method `upper`? (You don't need to have studied `upper` to answer this question about string method calls!)

A. `diss_upper`

B. `upper(diss)`

C. `diss.upper()`

D. `upper.diss()`

E. `str(upper(diss))`

(1) 22. The module `jedi` has a function named `info`, that takes a string (the name of a Jedi) as an argument, and returns a list of facts about that Jedi. Which of the following is a correct call to the function `info`, assuming that the import statement looks like this:

```
import jedi as jedi
```

(You don't need to have studied `jedi` to answer this question about using a module!)

A. `jedi.info("Rey")`

B. `info.jedi("Rey")`

C. `info("Rey")`

D. `"Rey".info()`

E. `jedi(info("Rey"))`

(1) 23. After the following assignment statement, what value does the variable `x` refer to?

```
x = print(str('ing'))
```

A. `'ing'`

B. `None`

C. `string`

D. `'string'`

E. `'print(ing)'`

Assume that the following variable declarations and initializations are given for questions 25 — 29.

```
example = "mbeoststalgee"
```

(1) 24. Which one of the following expressions has the string `'be'` as its value?

A. `example[0] + example[len(example)]`
B. `example[0] + example[len(example) - 1]`
C. `example[0:len(example) - 1]`
D. `example[1] + example[len(example) - 1]`
E. `example[1:len(example) - 1]`

(1) 25. Which one of the following strings is the value of the expression `example[5:10]`?

A. `'ststalg'`
B. `'oststal'`
C. `'tsta'`
D. `'tstalg'`
E. `'tstal'`

(1) 26. Which one of the following strings is the value of the expression `example[0:len(example):2]`?

A. `'mosle'`
B. `'message'`
C. `'bottle'`
D. `'bstg'`
E. `'mbeoststalgee'`

(1) 27. Which one of the following expressions has the string `'etae'` as its value?

A. `example[2:len(example):3]`
B. `example[2:len(example):5]`
C. `example[3:len(example):5]`
D. `example[3:len(example):4]`
E. `example[-2:0:-6]`

(1) 28. How would you slice the string `example` to produce the value: `'eeglatstsoeb'`

A. `example[len(example):0]`
B. `example[len(example):0:-1]`
C. `example[0:len(example):-1]`
D. `example[0:len(example)]`
E. `example[0:len(example):0]`

Part II — Written Answers.

Answer each question in the space provided on this question paper.

Section 1: Creating Functions

(2) 29. Write a Python function called `stringConcat` which has two string parameters: `s1` and `s2`. It should **return** a new string where `s1` is joined together (concatenated) with `s2` with a space added in between them. For example, if your function is called with the arguments `"Bob"` and `"Saget"`, it should return the string `"Bob Saget"`.

Section 2: Conditional Branching

(4) 30. Write Python code that obtains an integer from the console using the `input()` function, then displays on the console a message using the `print()` function indicating whether or not the integer is greater than or equal to 42.

(6) 31. Suppose three variables named `p1_score`, `p2_score`, and `p3_score` have already been created and initialized with numbers representing the scores in a game for player 1, player 2, and player 3, respectively. Write a python program that prints `Player 1 wins` to the console if player 1 had the highest score, `Player 2 wins` if player 2 had the highest score, `Player 3 wins` if player 3 had the highest score, or `It's a tie!` if two or more players are tied for the highest score (you **do not** have to say **which** two players tied).

Section 3: Loops

(3) 32. Mr. Burns has just informed Lenny that he will be fired if he cannot explain why he should keep his job without using the letter `'e'`.

Write Python code that reads a string from the console using the `input()` function and continues reading additional strings until the user enters a string that contains the letter `'e'`. You do not have to do anything with the strings that are read, just read strings until you get one that contains an `'e'`.

Hint: You can check whether a string contains a specific letter using the **`in`** operator, e.g. the value of the expression `'e' in s` (where `s` is a string) is `True` whenever `s` contains an `'e'`.

(4) 33. Suppose you have a list of strings called `teams` storing the names of teams in a sports tournament. The tournament is a round-robin tournament in which each team has to play each other team exactly once. Write a loop that prints out a complete list of all of the team matchups for the games that must be played; that is, each line of output must be `Lions vs. Oilers` where `'Lions'` and `'Oilers'` are team names from `teams`.

Hint 1: A team does not play itself.
Hint 2: Each match-up should only be printed **once**. So if you've already printed `Lions vs. Oilers`, you should **not also** print `Oilers vs. Lions`.

Section 4: Lists

(4) 34. Suppose you have two variables `list1` and `list2` that refer to lists of numbers, and that each list is the same length. Write code that generates a new list such that the *i*-th item in the new list is either the *i*-th item in `list1` or the *i*-th item in `list2`, whichever is larger. For example, if we had the lists `[1,3,7,2]` and `[2,0,3,9]`, your code would produce the list `[2,3,7,9]`.

(4) 35. In this question we will define the concept of a task as a list of two items: a string (the description of the task), and an integer between 1 and 10 indicating how important the task is. For example, `['Find Hairbrush', 3]` is a task.

Suppose you have a variable `tasks` that refers to a list of tasks, i.e., each item in `tasks` is a task as defined above. Write Python code that does two things. It produces a new list of tasks that contains only the urgent tasks that have an importance of 8 or more, and in the new list, each task is re-ordered so that the task importance is first, then the task description. **Use a loop-based solution.** Don't use list comprehensions (see the next question).

Example: if `tasks` is the list:

```
[ ['Find Hairbrush', 3], ['Find Food and Water', 10], ['Find Shelter', 9] ]
```

then the expected output would be

```
[ [10, 'Find Food and Water'], [9, 'Find Shelter'] ]
```

(2) 36. Write a solution to the previous question using a single list comprehension, but **don't** reverse the order of the sublists.

(2) 37. Suppose that a variable `prices` refers to a list of floating-point numbers that are the prices of the items in a customers shopping cart on an online store website. Write a **list comprehension** that creates a new list of prices that have had a 15% sales tax added to each original price.

Section 5: Dictionaries

(3) 38. Write a dictionary literal that associates the keys in the left column with the values in the right column:

`'vampires'`	10
`'ninjas'`	20
`'pirates'`	30

(4) 39. Suppose we are given a dictionary called `scenes` whose keys are pairs (tuples) of strings. The strings in each key are names of two characters from a movie. The values in the dictionary associated with the keys are Boolean values indicating whether the two characters named in the key appeared in a scene together.

An example dictionary entry in `scenes` might be:

```
('Frodo', 'Sam') : True
```

indicating that Frodo and Sam appeared in a scene together.

Given the dictionary variable `scenes` as described above, write python code that determines and prints out how many characters Frodo appears in a scene with.