15/22 Questions Answered
Saved on Nov 10 at 1:00 PM

TIME REMAINING
0:00

Final Exam (2021/S2) Multi-Choice and T/F Questions

STUDENT NAME

Search students by name or email...

Q1 Acknowledgment
0 Points

Australian National University

COMP1100 Final Exam, Semester 2 2021

You must acknowledge the following **integrity pledge** before proceeding. Please read carefully and check all the boxes.

- [ ] I am committed to being a person of integrity.
- [ ] I pledge, as a member of the ANU community, to abide by and uphold the standards of academic integrity outlined in the ANU statement on honesty and plagiarism, I am aware of the relevant legislation, and understand the consequences of breaching those rules.
- [ ] I will not communicate in any way with anyone else during this exam. This includes asking questions in any online forum.
- [ ] I acknowledge that this exam is protected by copyright and that copying or sharing any of its content will violate that copyright.

Read and check off the following instructions:

1. This examination is timed.

- [ ] Note the remaining time at the top right of this screen. Set an alarm for yourself if you need one.

2. Permitted materials. This is an open book exam. You might in particular find the course Website, the Prelude documentation, and the Data.List documentation useful.

- [ ] You may use any documentation you wish but **all work must be your own**.

Save Answer

Q2 Style

2 Points

Which of the following is False in Haskell?

- ○ Function names should reflect the intended use of the function
- ○ Using white box testing, you can test helper functions
- ◉ Function names should be kept as short as possible
- ○ Names of modules help readability of the code
- ○ Testing at the ghci prompt is volatile

✔ **Correct**

Save Answer Last saved on **Nov 10 at 12:58 PM**

Q3 Lists and Recursion

2 Points

Which of the following is False in Haskell?

- ◉ The length of the infinite list can be computed
- ○ Recursive functions are defined in terms of themselves
- ○ `[a]` can represent a list of functions
- ○ The pattern `x:[]` represents a list with only one element
- ○ `["COMP1100"]` is a singleton list

✔ **Correct**

Save Answer Last saved on **Nov 10 at 12:58 PM**

Q4 Type classes

2 Points

Which of the following is False in Haskell?

- ○ When a type is an instance of `Eq`, it means that we can use equality function over the type
- ○ `Double` type is an instance of the `Eq` typeclass
- ○ The built-in function `id` is polymorphic
- ◉ Ad-hoc polymorphism refers to functions with some inputs or outputs that could be of any type
- ○ By `Deriving` type classes, we can use default definitions of type class functions

✔ **Correct**

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q5 Higher Order Functions

2 Points

Suppose that you have a function `foo` with type

```
Double -> String -> String
```

Suppose that `foo` can in general run without producing errors. Which is False?

- ○ `foo 0` is a function that takes a `String` as input
- ○ `foo 0 "foo"` is type `String`
- ○ Running `foo 0` in ghci raises an error because function types are not instances of the `Show` typeclass
- ◉ `foo` takes a function of type `Double -> String` as input
- ○ `foo` is a function that takes a `Double` and returns a function that takes `String`

✔ **Correct**

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q6 Games

2 Points

Which of the following statement is False?

- (●) A heuristic function finds exact solutions for a problem
- ( ) Alpha-beta pruning and Minimax algorithm returns the same solutions given enough time
- ( ) Minimax algorithm starts from leaf nodes of the tree and propagates up the tree
- ( ) In the worst case, the time complexity of Alpha-beta pruning is the same as Minimax algorithm
- ( ) Pruned branches cannot be visited later

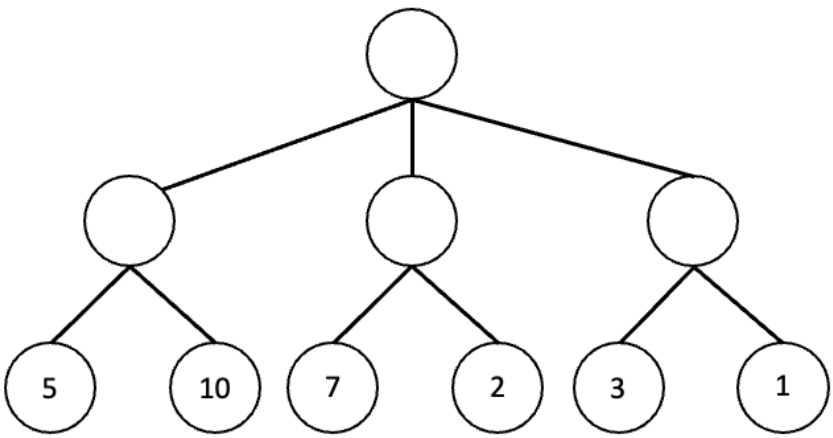
✔ Correct

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q7 Alpha-Beta Pruning

2 Points

Suppose that you are playing a two player game with alternating turns, and are using the alpha-beta pruning algorithm with lookahead 2, and it is your turn. Suppose the complete game tree looks as follows, with the values your heuristic would calculate indicated on the bottom level:



Which of the leaves of the tree, if any, will your algorithm prune (not visit)? (Explore children from left to right)

- ○ The leaves with values 2, 3, and 1.
- ○ The leaves with values 3 and 1.
- ○ The leaves with values 2.
- ◉ The leaves with values 1.
- ○ It will not prune any leaves.

✔ Correct

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q8 Tree

6 Points

Each correct answer **gains** you 2 marks, each incorrect answer **loses** you 1 mark, while a question left unanswered neither loses nor gains marks. The minimum total mark for this question is 0.

Consider the following definition:

```
data BinaryTree a = Null |
                    Node (BinaryTree a) a (BinaryTree a)
```

Select True if the follwoing is a **binary search tree** and False otherwise.

Q8.1 Tree

2 Points

```
Node (Node (Node Null 1 Null) 3 (Node Null 2 Null)) 5 (Node Null 6 
```

- ☐ True
- ✔ False

✔ Correct

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q8.2 Tree
2 Points

```
Node Null 1 (Node Null 2 (Node Null 3 (Node Null 4 (Node Null 10 Nu
```

- [x] True
- [ ] False

✔ Correct

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q8.3 Tree
2 Points

```
Node (Node (Node Null 1 Null) 2 Null) 3 (Node (Node (Node Null 4 Nu
```

- [x] True
- [ ] False

✔ Correct

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q9 Data Types
4 Points

Each correct answer **gains** you 2 marks, each incorrect answer **loses** you 1 mark, while a question left unanswered neither loses nor gains marks. The minimum total mark for this question is 0.

Consider the following Haskell definition.

```haskell
data Temperature = Low | Normal | High
type Temp = Int

tempToTemperature :: Temp -> Temperature
rangeTemperature :: Temperature -> [Temp]
```

Suppose that `tempToTemperature` and `rangeTemperature` are defined and can in general run without producing errors.

Q9.1 Data Types
2 Points

In ghci, `tempToTemperature 8` will return `Low`.

- [ ] True
- [x] False

✔ **Correct**

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q9.2 Data Types
2 Points

It is necessary to add `deriving Eq` to implement `rangeTemperature`.

- [ ] True
- [x] False

✔ **Correct**

Save Answer Last saved on **Nov 10 at 12:59 PM**

Q10 Complexity
4 Points

Consider the following function and select its time complexity.

Select one correct answer for each question. Correct answers receive full points. Incorrect answers receive no points.

```
-- | sort'
-- Given a list,
-- returns the sorted list in ascending order.

sort' :: (Ord a) => [a] -> [a]
sort' list = case list of
  [] -> []
  (x:xs) -> sort' sx ++ [x] ++ sort' lx
    where sx = filter (<=x) xs
          lx = filter (>x) xs
```

Q10.1 Complexity

2 Points

What is the **best case** time complexity of `sort'`?

- ○ $O(1)$
- ○ $O(\log n)$
- ◉ $O(n \log n)$
- ○ $O(n)$
- ○ $O(n^2)$

✔ Correct

Save Answer Last saved on **Nov 10 at 1:00 PM**

Q10.2 Complexity

2 Points

What is the **worst case** time complexity of `sort'`?

- ○ $O(1)$
- ○ $O(\log n)$
- ○ $O(n \log n)$
- ○ $O(n)$
- ◉ $O(n^2)$

✔ Correct

Save Answer Last saved on **Nov 10 at 1:00 PM**

Q11 Complexity

4 Points

Consider the following function and select its time complexity.

Select one correct answer for each question. Correct answers receive full points. Incorrect answers receive no points.

```
insert :: Ord a => a -> [a] -> [a]
insert x [] = [x]
insert x (y:ys)
   |x<=y = x:y:ys
   |otherwise = y: insert x ys

sort'' :: Ord a => [a] -> [a]
sort'' [] = []
sort'' (x:xs) = insert x (sort'' xs)
```

Q11.1 Complexity

2 Points

What is the **best case** time complexity of the Prelude function `sort''`?

- ○ $O(1)$
- ○ $O(\log n)$
- ○ $O(n \log n)$
- ◉ $O(n)$
- ○ $O(n^2)$

✔ **Correct**

Save Answer Last saved on **Nov 10 at 1:00 PM**

Q11.2 Complexity

2 Points

What is the **worst case** time complexity of `sort''`?

- ○ $O(1)$
- ○ $O(\log n)$
- ○ $O(n \log n)$
- ○ $O(n)$
- ◉ $O(n^2)$

✔ Correct

Save Answer Last saved on **Nov 10 at 1:00 PM**

Q12 Programming Questions

70 Points

There are **six** programming questions that you need to complete and submit.
You can find all programming questions on your dashboard.

Please submit by uploading **each** Haskell file to **each** question.

Please download the template Haskell files here.

Q12.1 Movies.hs

15 Points

Submit `Movies.hs` here

Save Answer

Q12.2 Numbers.hs

15 Points

Submit `Numbers.hs` here

Save Answer

Q12.3 Lists.hs

15 Points

Submit `Lists.hs` here

Save Answer

Q12.4 Trees.hs

10 Points

Submit `Trees.hs` here

Save Answer

Q12.5 Drop.hs

5 Points

Submit `Drop.hs` here

Save Answer

Q12.6 Instances.hs

10 Points

Submit `Instances.hs` here

Save Answer

Save All Answers

Submit & View Submission >