Computer Science Qualifying Exam
AM Session
Practice Exam
Computer Science Department, Reed College

You have two hours to complete this portion of the exam. There are five problems on 9 pages. Write your answers on the blank portions of each sheet, and on extra blank pages if needed. Make sure that your name and the problem number of your answer appear clearly on each sheet.

Complete each problem to the best of your ability. When the problem requests that you devise a program or a fragment of a program's code in a certain programming language you should do your best to produce working code in that language – syntactically correct and runnable – that meets the specification of the problem.

When you are providing code answers, if you you forget some aspect of the language, you can communicate your intended meaning for partial credit. As an example, if you forget how logical conjunction works in Python you could say, "I am using && to mean logical conjunction." You may find it useful to include explanations or comments in your answers. These are not necessary to achieve full credit, but may allow us to give more partial credit in the event of mistakes or things that are unclear.

Note that you are welcome to write additional functions that support the code you are asked to write, should you feel inclined to do so. (If the instructions ask you to "Write a function that..." that does not mean you cannot write some "helper" functions, as well.)

If you are unclear on what the problem is asking, you may ask a member of the Reed Computer Science faculty for clarification. If clarification is unavailable, please write out any assumptions you are making regarding the problem. You may not rely on any other external resources. Doing so is a violation of the honor principle.

AM1. [12 points] Alice owns a company that paints houses. The company just received an order to paint a group of houses all with the same color of paint. She is trying to figure out how much paint to purchase. She knows the area that a single can of paint covers, and she was given a list containing the area of the houses to be painted along with how many houses have that area. She wants to know how many cans of paint to order.

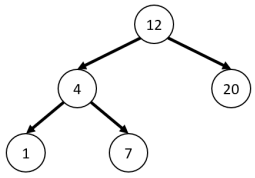
Write a Python function `paintCans` that takes an integer for the area a can of paint covers as the first parameter. The second parameter is a list of lists of integers giving the area of each house that needs to be painted. Each internal list has two values, the area of one group of houses, and the number of houses in that group. It should return the (integer) minimum number of paint cans Alice needs to complete the order.

AM2. [12 points] Below are two Python classes that we use to define a binary search tree. The tree defines an `insert` method that relies on a helper method `put` which places a key into the tree.

```
class TreeNode:
    def __init__(self,key):
        self.key = key
        self.parent = None
        self.left = None
        self.right = None

class BinarySearchTree:
    def __init__(self):
        self.root = None

    def insert(self, key):
        if self.root is None:
            self.root = TreeNode(key)
        else:
            self.put(key, self.root)

    def put(self, key, currentNode):
        if key < currentNode.key:
            if currentNode.left is None:
                currentNode.left = TreeNode(key)
                currentNode.left.parent = currentNode
            else:
                self.put(key, currentNode.left)
        else:
            if currentNode.right is None:
                currentNode.right = TreeNode(key)
                currentNode.right.parent = currentNode
            else:
                self.put(key, currentNode.right)
```

The interaction below builds the binary search tree pictured just to the right:

```
>>> t = new BinarySearchTree()
>>> t.insert(12)
>>> t.insert(4)
>>> t.insert(7)
>>> t.insert(1)
>>> t.insert(20)
```



Recall the terminology that each node may have a left *child* and a right child. For example, the node containing 12 has two children. These are the nodes containing keys 4 and 20. And prior to inserting 1, the node containing 4 has one child, namely the node with key 7. Furthermore, all nodes but the root node have *parents*. The root node is the parent of the nodes containing 4 and 20.

Write a Python method `height` that is a member of the TreeNode class. `height` takes no additional parameters, and returns the height of the node. We define the height of a node with no children to be zero, and the height of a node with children to be one greater than the maximum height of its children. For example, in the tree above, the node 7 has height 0 and the node 12 has height 2.

Write your code on the next page.

Write your code for the `height` function here.

AM3. [12 points] Below is the Python definition for a `MessageBoard` class. It represents a board that will display messages in a readable format on a character-limited screen. The method `display` displays the stored message in a character-limited format. The `changeMessage` function changes the message being displayed. It takes as input a string, and stores it as a number of chunks, where each chunk has a number of characters that is at most the size of the board.

```
class MessageBoard:
    def __init__(self, board_size):
        self.board_size = board_size            # Board size in characters
        self.message_chunks = []                # Board starts with no message
    def display(self)
        for chunk in self.message_chunks:
            print(chunk)
    def changeMessage(self, newMessage):
        message_chunks = []
        remaining = newMessage
        while(len(remaining) > 0):
            chunk_length = min(self.board_size, len(remaining))
            message_chunks = message_chunks + [remaining[:chunk_length]]
            remaining = remaining[chunk_length:]
```

Write the code for a `RentableMessageBoard` class that can be added to the code above. These are message boards that always display chunks of 256 characters at a time, but they also have a cost to display and a cost to load new messages. The costs are paid using funds loaded into the board. The `RentableMessageBoard` constructor should take two arguments: the cost (in dollars per message chunk) to change the message the board displays, and the cost (in dollars) of displaying the message. The constructor should also initialize the funds on the board to $0. In addition, the `RentableMessageBoard` should have a function `addFunds` that takes as an argument an amount of funds and adds that to the board's currently available funds. A `RentableMessageBoard` also overrides the two methods of a standard message board. In particular, `display` only works if the board has enough funds to cover the cost, and returns the funds spent to perform the action (which will either be the display cost, or 0 if not enough funds are available). Similarly, the `changeMessage` method depends on the available funds, with each message chunk costing the specified amount. If a `RentableMessageBoard` does not have enough funds to cover the entire message, only the chunks that have sufficient funds are stored (and charged for). The function should also return the amount of funds spent to change the message. The old message should be deleted even if there are not enough funds for a single message chunk.

Write your code on the next page.

Write your code for the `RentableMessageBoard` class on this page.

AM4. [12 points] Give the asymptotic worst-case running time for each of the following functions. Write this using Theta notation, fully simplified. (For example, $\Theta(n^2)$ or $\Theta(n \log n)$.) For each function, the input is a list and we define the size of the input list to be n.

(a) [3 points]

```
def functionA(list):
    a = 0
    for b in list:
        for c in list:
            if b + c >= 0:
                a = a + 1
            else:
                a = a - 1
    return a
```

(b) [3 points]

```
def functionB(list):
    g = 0
    for h in list:
        g = g + h
    i = 1
    j = 1
    while i < len(list):
        j = j * list[i]
        i = i * 2
    return g - j
```

(c) [3 points]

```
def functionC(list):
    for m in range(0,len(list)):
        list[m] = list[m] + 2
    return functionA(list)
```

(d) [3 points]

```
def functionD(list):
    u = 0
    for v in range(0,len(list)):
        list[v] = list[v] * -1
        sort(list)        # Assume a standard merge sort
    return functionB(list)
```

AM5. [12 points] Below are two Python classes that we use to define a doubly-linked list:

```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None
        self.prev = None

class LinkedList:
    def __init__(self):
        self.first = None

    def prepend(self, value):
        newNode = Node(value)
        newNode.next = self.first
        if self.first is not None:
            self.first.prev = newNode
        self.first = newNode

    def output(self):
        current = self.first
        while current is not None:
            print(current.value)
            current = current.next
```

In the code's use below, we build a linked list with `712` stored both at positions 0 and 1, `8` stored at position 2, `18` stored at position 3, and `37` stored at position 4.

```
>>> ll = LinkedList()
>>> ll.prepend(37)
>>> ll.prepend(18)
>>> ll.prepend(8)
>>> ll.prepend(712)
>>> ll.prepend(712)
>>> ll.output()
712
712
8
18
37
```

Write a method `reverse` that modifies the list so that the values are in the opposite order they started in. For full credit, you should not modify the `value` of any `Node`.

Put your Python code for `reverse` on the next page.

Write your code for the `reverse` function on this page.

Computer Science Qualifying Exam
PM Session
Practice Exam
Computer Science Department, Reed College

You have two hours to complete this portion of the exam. There are five problems on 7 pages. Write your answers on the blank portions of each sheet, and on extra blank pages if needed. Make sure that your name and the problem number of your answer appear clearly on each sheet.

Complete each problem to the best of your ability. When the problem requests that you devise a program or a fragment of a program's code in a certain programming language you should do your best to produce working code in that language – syntactically correct and runnable – that meets the specification of the problem.

When you are providing code answers, if you you forget some aspect of the language, you can communicate your intended meaning for partial credit. As an example, if you forget how logical conjunction works in Python you could say, "I am using && to mean logical conjunction." You may find it useful to include explanations or comments in your answers. These are not necessary to achieve full credit, but may allow us to give more partial credit in the event of mistakes or things that are unclear.

Note that you are welcome to write additional functions that support the code you are asked to write, should you feel inclined to do so. (If the instructions ask you to "Write a function that..." that does not mean you cannot write some "helper" functions, as well.)

If you are unclear on what the problem is asking, you may ask a member of the Reed Computer Science faculty for clarification. If clarification is unavailable, please write out any assumptions you are making regarding the problem. You may not rely on any other external resources. Doing so is a violation of the honor principle.

PM1. [12 points] Write a C++ function `chooseBit`. This is a function that takes as arguments an unsigned 8-bit integer value (of type `uint8_t`) and an integer between 0 and 7 and returns the bit at the specified position in the 8-bit value. The return value should be either a 1 or a 0. For example, if the bits of the first input are defined as X7,X6,X5,X4,X3,X2,X1,X0 and the second value is 3, then the return value should be X3.

PM2. [12 points] Write a Python function `greater`. This is a function that returns another function. It should take an integer as its argument. It should return a function that takes an integer argument and returns whether or not the argument is greater than the original.

```
>>> x = greater(12)
>>> x(20)
True
```

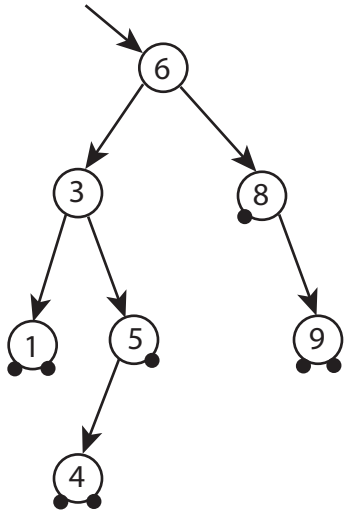
PM3. [12 points] Write a C++ function with the prototype:

```
void deallocate(uint32_t** matrix, uint32_t length, uint32_t width);
```

The function takes as arguments a pointer to a two-dimensional matrix of integers, a length value, and a width value. The length value shows the size of the first dimension, and the width shows the size of the second dimension. The function should deallocate the entirety of the matrix pointed to by the first argument.

PM4. [12 points] The C++ code for binary search tree node is given below left and a picture of a binary search tree is shown to the right:

```
struct node {
    uint32_t data;
    struct node* parent;
    struct node* left_child;
    struct node* right_child;
};
```



Write a C++ fuction `insert` for inserting a value into the binary search tree. The function should take as an argument a pointer to the root node of the tree and a value to add into the tree. It should create a new node containing the value and add it into the input tree as a leaf in the appropriate location according to the binary search tree property. For this problem, we will make two assumptions. The first is that the input tree will never be empty. The second is that the tree is not allowed to contain duplicates. Preventing duplicates means that if the value already exists in the tree, the function should not create a new node and should return `false`. If the value does not exist, a new node should be created and the function should return `true`.

PM5. [12 points] Using only the MIPS32 instructions on the guide in the bottom half of this page, write MIPS assembly code that determines whether a given value can be found in the elements pointed to by an array of pointers. Remember that pointers in MIPS are 32 bits.

Preconditions: Your code can assume that

- Elements in the array are valid 32-bit pointers that each point to a 32 bit unsigned integer.
- Register `$a0` gives the value being searched for (a 32 bit unsigned integer).
- Register `$a1` gives the (word-aligned) address of the first element in the array.
- Register `$a2` gives the length of the array.

Postconditions: Upon completion

- Register `$v0` contains the index of the first element that points to a value matching the input, or -1 otherwise.

Having completed its work, your code should jump to a label `done`. You can use any of the registers `$t0` through `$t9` to perform your calculations.

Please write your MIPS code on the next page, using only instructions from those listed below.

MIPS32 Assembly Guide

`li` $RD, *value*	loads an immediate value into a register
`lw` $RD, ($RS)	loads a register from memory at an address specified in a register
`sw` $RS, ($RD)	stores a register into memory at an address specified in a register
`addu` $RD, $RS1, $RS2	add two registers, storing the sum in another
`subu` $RD, $RS1, $RS2	subtract two registers, storing the difference in another
`addiu` $RD, $RS1, *value*	add a value to a register, storing the sum in another
`move` $RD, $RS	copy a register's value to another
`j` *label*	jump to a labeled line
`blt` $RS1, $RS2, *label*	jump to a labeled line if one register's value is less than another
`bltz` $RS, *label*	jump to a labeled line if a register's value is less than zero
`gt`, `le`, `ge`, `eq`, `ne`	other conditions than `lt`
`sll` $RD, $RS, *value*	shifts the value in the source register *value* bits left
`sllv` $RD, $RS1, $RS2	shifts the value in a source register another source register bits left

Write your MIPS code here:

Computer Science Qualifying Exam
Discrete Mathematics (DM) Session
10am - 12pm, March 5, 2023
Computer Science Department, Reed College

You have two hours to complete this portion of the exam. There are four problems on 6 pages. Write your answers on the blank portions of each sheet, and on extra blank pages if needed. Make sure that your name and the problem number of your answer appear clearly on each sheet.

Complete each problem to the best of your ability. When a problem asks you for a formal proof or justification, please be thorough and follow the standards that you have been taught in class. A final answer is not enough for full credit for most problems.

For problems that ask for a numeric answer, you should simplify your answer as much as reasonably possible. We have provided a basic calculator so that you can add, subtract, multiply, and divide to compute your numeric answers. We recommend you show what calculations you are doing for clarify and partial credit.

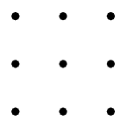
If you are unclear on what the problem is asking, you may ask a member of the Reed Computer Science faculty for clarification. If clarification is unavailable, please write out any assumptions you are making regarding the problem. You may not rely on any other external resources. Doing so is a violation of the honor principle.

DM1. [4 points each, 24 total] Below are several short questions. Write your answer clearly in the box provided. We have provided extra space if you need to do work, but there is no partial credit for these problems — if you know the answer, there's no need to write out a long explanation. When it asks for a number, please give a specific number. (For example, don't write "5!", write "120". You have a calculator if you need it.)

(a) One fair die has faces 1, 1, 2, 2, 3, 3 and another has faces 4, 4, 5, 5, 6, 6. The dice are rolled and the numbers on the top faces are added. What is the probability that the sum will be odd?

(b) We shuffle a deck of 10 cards, numbered 1 through 10, so that all orderings are equally likely. We then pick up to the top four cards. What is the probability that those cards are in order from least to greatest?

(c) A set of three points is chosen randomly from the grid shown. Each three-point set has the same probability of being chosen. What is the probability that the points lie in a straight line?



(d) A casino offers a simple dice game. A player picks a number between 1 and 4, then rolls two (fair) 4-sided dice. If one die rolls the number the player guessed, they win a dollar. If two dice roll that number, they win two dollars. If neither die does, then they lose a dollar. Is it smart to play this game? What is the average gain/loss for the player?

(e) How many numbers in $\{1, 2, 3, \ldots, 1999\}$ are integer multiples of 3 or 4 but not 12?

(f) Four numbers, a, b, c, and d are chosen at random from the set $\{0, 1, 2, 3, \ldots, 1999\}$. (The numbers are chosen uniformly and independently. There is no requirement that they be distinct.) What is the probability that $ab - cd$ will be an even number?

DM2. [12 points] Prove that for any integers a and b we have

$$\gcd(a^5, b^5) = \gcd(a, b)^5$$

where gcd denotes the greatest common divisor. (Hint: One good solution involves prime factorization.)

DM3. [12 points] Let p be a prime number greater than 5. Show that $p^2 - 1$ must be divisible by 24. (Hint: Consider the values of p^2 mod 3 and mod 8.)

DM4. [12 points] Let $S = \{2, 3, 5, 7, 11, 13, 17, 19\}$ be the set of prime numbers less than 20. For a subset $A \subseteq S$, define the sum of S to be $\text{sum}(A) := \Sigma_{k \in A} k$. Prove that there are four nonempty subsets of S with the same sum. (Hint: what is the largest possible $\text{sum}(A)$ for $A \subseteq S$? What is the minimum $\text{sum}(A)$?)