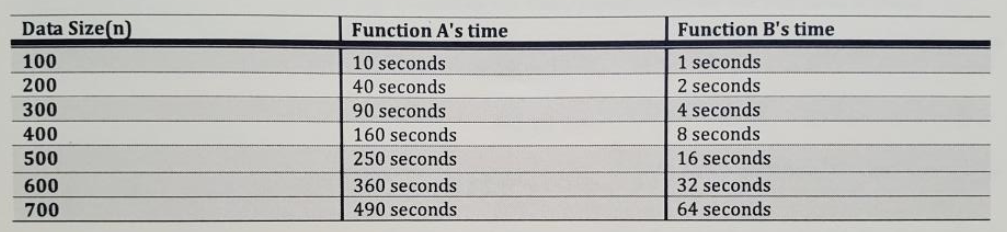
**Question 1 [4marks]: big-O**

Suppose there were two functions A and B. both functions will accomplish the same task but seem to run with different times. The chart below shows the amount of time it took for each function to complete the task on various amount of data.



1. If the pattern of time continues, what is the likely run time (in big-O notation) for function A? (1 mark)
2. If the pattern of time continues, what is the likely run time (in big-O notation) for function B? (1 mark)
3. Under what circumstances would you use function A? When would you use function B? (2 mark)

**Question 2 [4marks]: queues**

Explain how to implement a Queue using an array such that its enqueue(), dequeuer() and front() function can all be accomplished in O(1) time. Provide details.

**NOTE: an explaination is not a code dump. Providing the code/pseuo-code for the functions without explainantion will earn 0 marks**

**Question 3 [4marks]: application of concept**

At an interview, an interviewer asks you following question:

“The build and test time for our application is 30 min. Yesterday, when we ran our tests, our program passes every test. Today, when we ran our tests, our program is failing one of our tests. Between yesterday’s testing and today testing there has been 100 commits. One of those commits introduced this bug. How would you go about finding which commit it was?”

1. **Explain how you would go about figuring out which version (which commit) had the bug in it.**

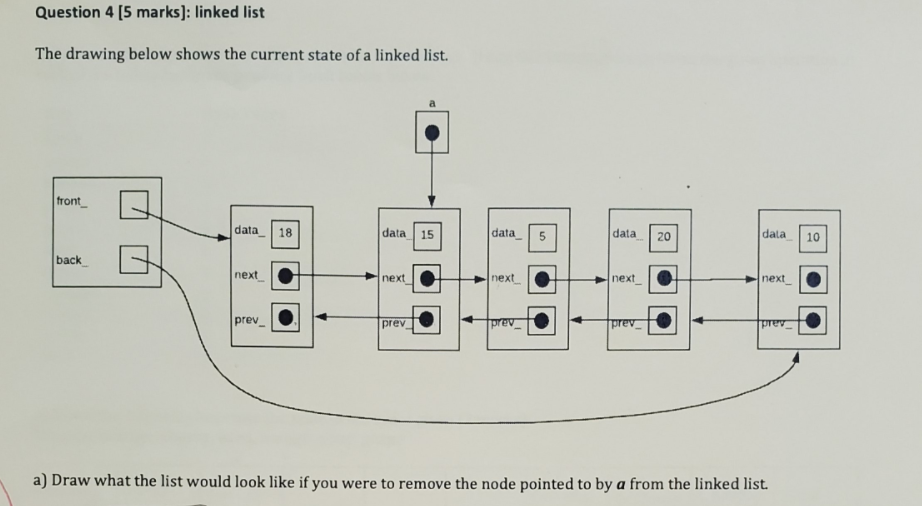
ANSWER:

1. **What is the maximum amount of time (number of minutes) it would take for you to figure out which commit has the bug?**

ANSWER:

Side Notes:

* **Build and test time** is the time it takes to compile and test the program. In order to test a version, you must build it then run the tests. Thus each version you test will take 30 minutes to determine if that particular version is passing testing.
* **Commits** refer to a version. 100 commits means 100 different versions. This is like a commit you make to your github repo.



1. Filler \*\*ignore this line
2. Does the following blurb of code correctly remove the node pointed to by **a**? If not, rewrite the code blurb so that it works.

a->prev\_->next\_ = a->next\_;

a->prev\_ = nullptr;

a->next\_->prev\_ = a->prev\_;

a->next\_ = nullptr;

delete a;

**Question 6 [7 marks]: analysis**

What is the run time of the function1() and function2(), and function3() with respect to n. Ensure that the analysis is clear and detailed.

int function1(int n){

int rc = 0;

for(int I = 1; i <= 10; i++){

rc += 1;

}

return rc;

}

int function2(int n){

int rc = 0;

for(int I = 1; i <= n\*n; i++){

rc += i;

}

return rc;

}

int function3(int n){

int rc;

rc = function2(n);

for(int i = 0; i < n; i++){

rc += function1(n);

}

return rc;

}

**Question 7 [4marks]: Quicksort**

Quicksort has an average case O(n log n) run time but a worst case run time of O(n^2).

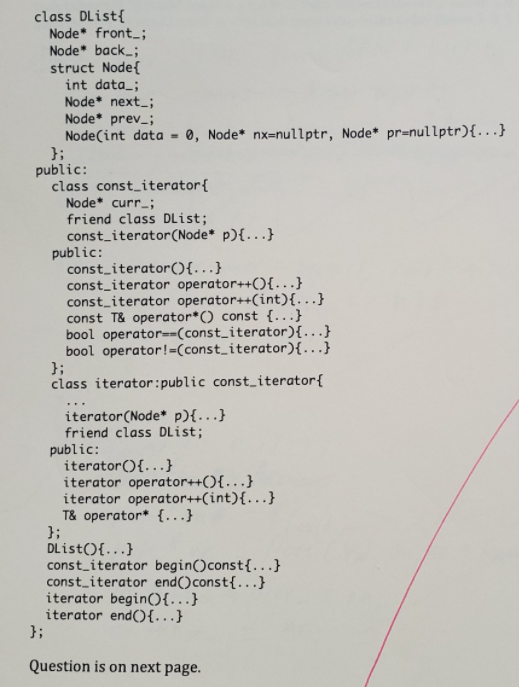
1. In 2 or 3 sentences, describe the circumstances under which quicksort would end up at its worst case run time of O(n^2).
2. In 2 or 3 sentences, describe one method that will help reduce the likelihood of the worst case run time being reached.

**Question 9 [12marks]: Linked Lists**

In this coding question for linked lists, you may assume that the list use sentinels or not use the sentinels. The choice is yours but you must pick: **if you do not pick, I will mark your code assuming the linked list does NOT using sentinels.**

**Will your code include the use of sentinels (circle one): YES or NO**

Suppose you were given the following declaration for a node and linked list. **Every function listed below is a function you can call without implementing.** Any function not listed must be coded by you.



1. Filler \*\*ignore this line
2. Write the following member function in O(1) time: (6 marks)

**Void Dlist::prepend(DList& other);**

This function prepends (puts nodes in other BEFORE the nodes in the current list) the list in other to the front of the current list ***other*** becomes an empty list.

Example:

Current list contains the nodes with following values in order given {1,5,2}.

***Other*** contains the nodes with following values in order given {6,3,9}.

This function makes the current list: {6,3,9,1,5,2} and other {}.