**CS 421 Programming HW1 (100 pts, October 25, 2021 by 11:59pm)**

**Name: Aman Patel**

In this programming projects, you will write a program to determine minimum number of scorings for any given integer n>1.

Documentation:

1. **(5 pts) appropriate files submitted for full credit with appropriate naming**

One zip file containing….

Documentation name: patel\_Aman\_project1\_doc.dox

Implementation: patel\_Aman\_project1.java

1. **(5 pts) What is input size of your solution?**

* Input size of my solution is log(n) since it is an integer greater than 1.

1. **(10 pts) Which data structure(s) have you used? How such structure(s) helped to obtain better run-time complexity**

* For this problem, I haven’t really used any data structures particularly, but I did used an idea of a heap structure.
* As in the max heap structure parent node is greater compared to child node,
* In our problem we must find the number of minimum possible scores to make a final total.
* To do that we have to see how many largest possible points scores are there, than we move to second largest and so on.
* We can create a heaps of different pointing score say first heap is 8-point score, then 7-point score and so on. And using the help of these heaps we can find the solution.
* But in my solution, it was easier to just use this idea and to represent them by numbers and fractions.

1. **(30 pts) What is the complexity of your solution? Justify it.**

* Complexity of my solution will be O (14) in worst case and O(5) in best case.

|  |  |
| --- | --- |
| int n; | 1 |
| Scanner input=new Scanner(System.in); | 1 |
| System.out.print("Enter a score: "); | 1 |
| n=input.nextInt(); | 1 |
| if (n<=1)print; | 1 |
| **Below are the conditions if numbers are between 2 and 9** | |
| else if(n<=9) { | 1 |
| if(n==2)print; | 1 |
| else if(n==3)print; | 1 |
| else if(n==4)print; | 1 |
| else if(n==5)print; | 1 |
| else if(n==6)print; | 1 |
|  | 1 |
| else if(n==8)print; | 1 |
| else if(n==9)print; } | 1 |
| **Below is the condition for all the multiples of 8. Except 8 itself.** | |
| else if(n%8==0) print; | 1 |
| **Below are the conditions for all the numbers like 8n-2, 8n-3, 8n-6 and 8n-7.** | |
| else if((n-2)%8==0)print; | 1 |
| else if((n-3)%8==0)print; | 1 |
| else if((n-6)%8==0)print; | 1 |
| else if((n-7)%8==0)print; | 1 |
| **Below are the conditions for all the numbers like 8n-4 and 8n-5.** | |
| else if((n-4)%8==0)print; | 1 |
| else if((n-5)%8==0)print; | 1 |
| **Below are the conditions for all the numbers like 8n-9 or 8n-1.** | |
| else if((n-9)%8==0) print; | 1 |
|  |  |
| Best case | **O(5)** |
| Worst case | **O(14)** |

* As shown in the above table best case scenario of the program will be O(5) when the score entered will be 1 or less than one.
* But if That is not the case, we have two different possibilities, and they are as follows.
  1. If the total score is between 2 and 9
  2. Else the total score is more than 9.
* Highlighted part here on the table is the conditions for total score between 2 and 9.
* And everything below is if the number is more than 9.
* Since I have used if and else if statements as soon as one of the conditions is true program will stop checking the rest of the files.
* if statements and else if statement has time complexity if O(1), as they are paired with simple print statement.