1. (8 points) Define $\Pi : \mathbb{N} \rightarrow \{0,1\}$ by $\Pi(i)$ equals 1 if there is a sequence of at least $i$ consecutive 5’s in the decimal expansion of $\pi$, and 0 otherwise. Is $\{i|\Pi(i) = 1\}$ recursive? Justify your response. By the way, we do not currently know if there exists an $i \in \mathbb{N}$ such that $\Pi(i) = 0$. Hint: What are the possible values of $\{i|\Pi(i) = 1\}$?

2. (10 points) Show that the following problem is undecidable:

Given Turing Machine $M$, state $q$ of $M$, and string $x \in \Sigma^*$, will $M$ with input $x$ ever enter state $q$? That is, is there an $n$ such that $(s, x, 0) \xrightarrow{\ast} (q, \alpha, n)$?

This problem is analogous to solving the dead code problem – you can’t always decide whether some code in a program can ever be executed?

3. (18 points) For each of the following questions, justify your answer. In your justification, you may assume that for any recursively enumerable set there is a TM to list it, and for any recursive set there is a TM to list it in lexicographic order.

Are recursively enumerable sets closed under
- union?
- intersection?
- complement?

Are recursive sets closed under
- union?
- intersection?
- complement?
1. \( \{ i | \Pi(i) = 1 \} \) is recursive. There are two possibilities, either \( \{ i | \Pi(i) = 1 \} = \mathbb{N} \), in which case it is accepted by a total TM that immediately enters the accept state. or there exists an \( n \) such that
\[
\Pi(i) = \begin{cases} 
1, & \text{if } i \leq n \\
0, & \text{if } i > n 
\end{cases}
\]
in which case there is a total TM that enters the accept state if the number on the input tape is less than or equal to \( n \), or else it enters the reject state. Although we don’t know which TM solves the problem, we don’t have to know. There exists a total TM to solve the problem, and \( \{ i | \Pi(i) = 1 \} \) is recursive.

2. The problem is not decidable. We can reduce the Halting Problem (known to be undecidable) to the problem of deciding if an arbitrary machine ever enters state \( q \).

We convert TM \( M \) to a new TM \( M' \) such that \( M' \) has a new state \( q \), and on any input \( M \) halts if and only if \( M' \) enters state \( q \). We change the transition function of \( M \) by:
\[
\delta(t,a) = (q,a,R), \quad \forall a \in \Gamma \\
\delta(r,a) = (q,a,R), \quad \forall a \in \Gamma \\
\delta(q,a) = (q,a,R), \quad \forall a \in \Gamma
\]
So \( M \) halts on input \( x \) if and only if \( M' \) ever enters state \( q \) on input \( x \). If we could decide whether an arbitrary TM with an arbitrary input ever enters a certain state, then we could solve the Halting Problem, which is undecidable.

3. If either of \( L_0 \) or \( L_1 \) is finite, we can list it before listing the other set. Hence we may assume that both sets are infinite. Assume that re sets \( L_0 \) and \( L_1 \) can be listed by Turing Machines \( M_0 \) and \( M_1 \) as \( y_0, y_1, ... \) and \( z_0, z_1, ... \) respectively.

**Union:** By running \( M_0 \) and \( M_1 \) in parallel, and waiting until each one in turn provides its next output, we can list the members of \( L_0 \cup L_1 = y_0, z_0, y_1, z_1, ... \). Hence, re sets are closed under union.

**Intersection:** We can run \( M_0 \) and \( M_1 \) in parallel, keeping lists of the elements as they appear. As each \( y_i \) appears, we check if it has appeared among the \( z_0, z_1, ..., z_j \) which have already been listed. If so, we output it as \( x_k \). As each \( z_j \) appears, we check if it has appeared among the \( y_0, y_1, ..., y_i \) which have already been listed. If so, we output it as \( x_k \). So every element which belongs to \( L_0 \cap L_1 \) will ultimately appear in both of \( y_0, y_1, ... \) and \( z_0, z_1, ... \), and will ultimately be listed. Hence, re sets are closed under intersection.
**Complement:** The set of re sets is not closed under complement. If it were, then every re set would be recursive, and the Halting Problem would be decidable. To see this, assume that for any re set \( L \), its complement \( \overline{L} \) were also re. Then we could list \( L \) and \( \overline{L} \), being assured that ultimately every string in \( \Sigma^* \) would appear in exactly one of the lists. So we could decide on membership in \( L \), and thus it would be recursive. But this is not true (a contradiction) because we know that there are re sets which are not recursive.

Recursive sets are closed under union, intersection and complement. For any recursive set \( L \) and any string \( x \in \Sigma^* \), we can decide if \( x \in L \). So we can decide on membership in the union, intersection or complement of recursive sets by using the appropriate one of:

\[
\text{if } x \in L_0 \text{ or } x \in L_1 \text{ then return } x \in L_0 \cup L_1 \text{ else return } x \notin L_0 \cup L_1 \\
\text{if } x \in L_0 \text{ and } x \in L_1 \text{ then return } x \in L_0 \cap L_1 \text{ else return } x \notin L_0 \cap L_1 \\
\text{if } x \in L \text{ then return } x \notin \overline{L} \text{ else return } x \in L
\]