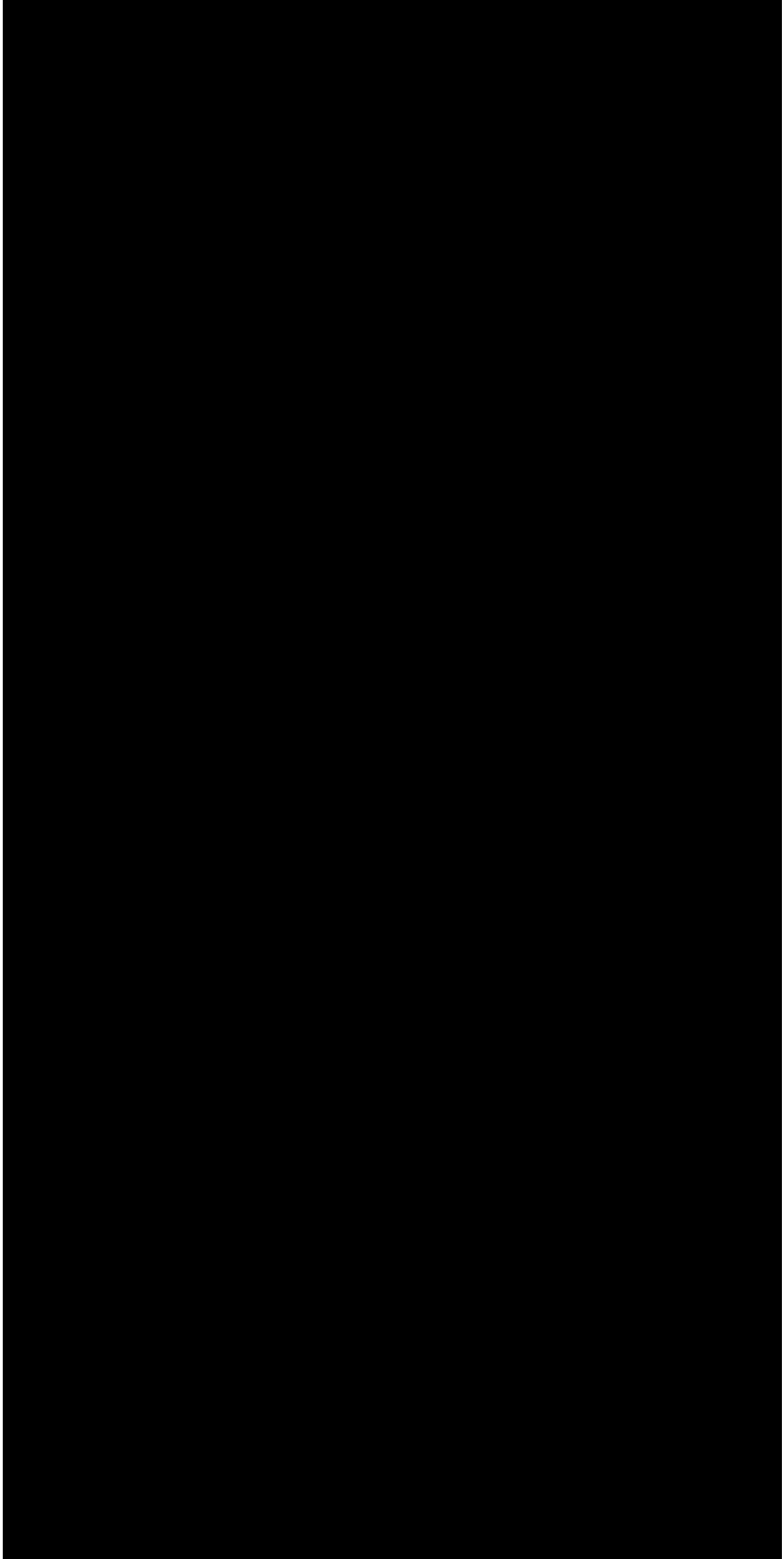
```
Enter the maximum number of hospitals that can be constructed: 4
Yes, can offer health care to all!
Hospital - 1
Hospital location: A
Hospital - 2
Hospital location: DF
Hospital - 3
Hospital location: BC
Hospital - 4
Hospital location: CED
```

```
Enter the maximum number of hospitals that can be constructed: 5
Yes, can offer health care to all!
Hospital - 1
Hospital location: A
Hospital - 2
Hospital location: DF
Hospital - 3
Hospital location: BC
Hospital - 4
Hospital location: CED
```

```
Enter the starting position (street and avenue): 2 3
Number of paths to return home: 3
```



```
Enter the starting position (street and avenue): 6 6
Number of paths to return home: 252
```

```
Enter the maximum number of hospitals that can be constructed: 3
No, some cities are not covered.
```

```
Seven of Spades
  Six of Diamonds
 Nine of Spades
Queen of Diamonds
 Jack of Diamonds
Three of Hearts
 Jack of Hearts
Three of Spades
 King of Spades
Seven of Hearts
Seven of Diamonds
Deuce of Hearts
 Four of Hearts
  Ten of Hearts
 Nine of Diamonds
Three of Clubs
  Six of Hearts
 King of Diamonds
  Ace of Clubs
  Ace of Diamonds
 King of Clubs
 Jack of Spades
 Five of Clubs
Queen of Hearts
 Five of Diamonds
  Ten of Clubs
 Four of Diamonds
 Nine of Hearts
 Jack of Clubs
 Five of Spades
Deuce of Spades
 Four of Spades
 King of Hearts
Queen of Spades
  Six of Clubs
 Nine of Clubs
  Ten of Diamonds
Queen of Clubs
Eight of Hearts
Deuce of Clubs
  Six of Spades
Deuce of Diamonds
Seven of Clubs
Eight of Clubs
  Ace of Hearts
 Five of Hearts
 Four of Clubs
Eight of Diamonds
Three of Diamonds
  Ten of Spades
```

```
  Ten of Spades
Deuce of Hearts
 Nine of Hearts
Queen of Diamonds
 King of Hearts
Eight of Spades
  Ten of Diamonds
Three of Hearts
  Ten of Clubs
 Four of Diamonds
 Jack of Diamonds
 Four of Hearts
  Six of Diamonds
  Ace of Clubs
 Five of Spades
  Six of Clubs
 Nine of Spades
 Four of Clubs
Queen of Clubs
Seven of Clubs
Queen of Spades
Eight of Diamonds
  Ten of Hearts
 Jack of Clubs
Three of Diamonds
Seven of Hearts
  Ace of Hearts
Eight of Clubs
  Six of Spades
 Five of Hearts
Queen of Hearts
 Nine of Diamonds
 King of Diamonds
  Ace of Spades
Seven of Spades
Deuce of Diamonds
 Four of Spades
Seven of Diamonds
 Jack of Spades
Three of Clubs
 Nine of Clubs
 Jack of Hearts
Eight of Hearts
 Five of Clubs
 King of Clubs
  Ace of Diamonds
 Five of Diamonds
  Six of Hearts
Deuce of Clubs
Deuce of Spades
 King of Spades
Three of Spades
```

In Part 1, I developed a recursive function numPathsHome to determine the number of possible paths for a robot courier (Amazon Scout) to return to the dispatch center from a given starting position. The robot can only move west (left) and south (down). The solution involves using a recursive approach to explore all potential paths from the starting point to the destination. The base cases include reaching the dispatch center and moving out of bounds. By summing the results of recursive calls for west and south movements, the function calculates the total number of paths.

In Part 2, I created a recursive function to determine if it is possible to provide healthcare coverage to all cities using a limited number of hospitals, given cost and coverage constraints. Each hospital is represented by a struct containing its name and the cities it can serve. The solution involves recursively exploring combinations of hospitals to see if all cities can be covered within the specified limit. If a valid combination is found, the result parameter is updated with the selected hospitals.

In Part 3, I wrote a program to shuffle a virtual deck of 52 cards and display the shuffled order. Each card is represented by a struct containing its face and suit. The solution includes creating the card structs, implementing a shuffling algorithm to randomly rearrange the cards in the deck, and displaying the shuffled deck in the specified format.

```
Enter the starting position (street and avenue): 3 4
Number of paths to return home: 10
```

```
Enter the starting position (street and avenue): 6 7
Invalid starting position!
```