```
class DoubleHashing:
    def __init__(self, size, prime):
        self.size = size
        self.keys = [None] * size
        self.collisions = [0] * size
        self.prime = prime

    def hash_func_1(self, key):
        return key % self.size

    def hash_func_2(self, key):
        return self.prime - (key % self.prime)

    def insert_dh(self, key):
        i = self.hash_func_1(key)
        i = i

        while self.keys[i] is not None:
            i = (i + self.hash_func_2(key)) % self.size
            self.collisions[i] += 1
            if i == i+1:
                print("Hash table is full. Cannot insert.")
                return

        self.keys[i] = key

    def search_dh(self, key):
        i = self.hash_func_1(key)
        i = i

        while self.keys[i] is not None:
            if self.keys[i] == key:
                return i
            i = (i + self.hash_func_2(key)) % self.size
            if i == i:
                break

        return None

    def display(self):
        print("| Index | Key | Collisions |")
        print("|-------|-----|------------|")
        for i in range(self.size):
            print(f"|   {i}   |  {self.keys[i]}  |     {self.collisions[i]}     |")



def display_table(table):
    for i, value in enumerate(table):
        print(f"i {i}: {value if value is not None else 'Empty'}")



def find_prime(size):
    p_no = size - 1 if size % 2 == 0 else size - 2
    while p_no >= 2:
        if is_prime(p_no):
            return p_no
        p_no -= 2
    return None



def is_prime(num):
    if num < 2:
        return False
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            return False
    return True



def main():
    while True:
        print("\nMain Menu")
        print("Select one of these options:")
        print("1) Double Hashing")
        print("2) Exit")

        menu_choice = int(input("Enter your choice: "))



        if menu_choice == 1:
            size = int(input("Enter the size of the hash table: "))
            prime_num = find_prime(size)

            if prime_num is None:
                print("Error: Unable to find a prime number less than the specified
table size.")
                continue

            print(f"Using {prime_num} as the prime number for double hashing.")
            double_table = DoubleHashing(size, prime_num)
            submenu(double_table, "Double Hashing")
        elif menu_choice == 2:
            break
        else:
            print("Invalid choice. Please enter a valid option.")
```

```
def submenu(hash_table, technique):
    while True:
        print(f"\n{technique} Menu")
        print("Select one of these options:")
        print("1) Insert")
        print("2) Search")
        print("3) Display")
        print("4) Return to Main Menu")

        choice = int(input("Enter your choice: "))

        if choice == 1:
            value = int(input("Enter the value to insert: "))
            hash_table.insert_dh(value)
        elif choice == 2:
            value = int(input("Enter the value to search: "))
            result = hash_table.search_dh(value)
            if result is not None:
                print(f"{technique}: Value found at i {result}.")
            else:
                print(f"{technique}: Value not found.")
        elif choice == 3:
            hash_table.display()
        elif choice == 4:
            break
        else:
            print("Invalid choice. Please enter a valid option.")



if __name__ == "__main__":
```

```
main()
```