```
class Job:
    def __init__(self, id, deadline, profit):
        self.id = id
        self.deadline = deadline
        self.profit = profit

def job_sequencing(jobs):
    # Sort jobs by profit in descending order
    jobs.sort(key=lambda x: x.profit, reverse=True)

    max_deadline = max(job.deadline for job in jobs)
    slots = [None] * max_deadline  # Create empty slots for each deadline
    total_profit = 0

    for job in jobs:
        for i in range(job.deadline - 1, -1, -1):  # Check from last slot to first
            if slots[i] is None:  # Find an empty slot
                slots[i] = job.id  # Assign job to slot
                total_profit += job.profit  # Add profit to total
                break  # Stop after finding the first available slot

    return [job_id for job_id in slots if job_id is not None], total_profit

def get_jobs():
    n = int(input("Enter number of jobs: "))
    jobs = []
    for _ in range(n):
        job_id = input("Job ID: ")
        deadline = int(input("Deadline: "))
        profit = int(input("Profit: "))
        jobs.append(Job(job_id, deadline, profit))
    return jobs

# Main logic
if __name__ == "__main__":
    jobs = get_jobs()  # Get job details from the user
    scheduled_jobs, total_profit = job_sequencing(jobs)  # Maximize profit by scheduling jobs
    print("Scheduled Jobs:", scheduled_jobs)  # Print the job IDs in the order they were scheduled
    print("Total Profit:", total_profit)  # Print the total profit
```