# CSCI 3104-Spring 2015: Assignment #7

**Assigned date:** Tuesday 3/10/2015,  
**Due date:** Tuesday, 3/17/2015, before end of class  
**Maximum Points:** 45 points + 5 for legibility.

**Note:** This assignment *must be turned in on paper, before class*. Please do not email: it is very hard for us to keep track of email submissions. Further instructions are on the class page: [http://csci3104.cs.colorado.edu](http://csci3104.cs.colorado.edu)

## P1 (25 points)
An investor has a budget of B dollars (assume B is a whole number) and considers a series of investments, each with a cost, payoff and a risk score, as shown below:

<table>
<thead>
<tr>
<th>ID</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>...</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>c_1</td>
<td>c_2</td>
<td>c_3</td>
<td>...</td>
<td>c_n</td>
</tr>
<tr>
<td>Payoff</td>
<td>p_1</td>
<td>p_2</td>
<td>p_3</td>
<td>...</td>
<td>p_n</td>
</tr>
<tr>
<td>Risk Score</td>
<td>r_1</td>
<td>r_2</td>
<td>r_3</td>
<td>...</td>
<td>r_n</td>
</tr>
</tbody>
</table>

1. Each investment ID $j$ has a cost of $c_j$, a payoff of $p_j$ and a risk score of $r_j$ (for convenience, $c_j, p_j, r_j$ assumed to be whole numbers).
2. The goal is to select a subset of the investments from 1 to $n$,
3. Investing a maximum of B dollars,
4. Limiting the total risk score to $R$,
5. While *maximizing* the total payoff.

Consider the recursive function $\text{payoff}(\hat{B}, \hat{R}, j)$ denoting

$$\text{payoff}(\hat{B}, \hat{R}, j): \text{the MAXIMUM payoff possible considering investments ID in the range 1 to } j, \text{ budget } \hat{B} \geq 0 \text{ and max risk } \hat{R} \geq 0.$$  

For the base cases,
1. set $\text{payoff}(\hat{B}, \hat{R}, 0) = 0$ for any $\hat{B} \geq 0$, $\hat{R} \geq 0$, and
2. set $\text{payoff}(\hat{B}, \hat{R}, j) = -\infty$ if $\hat{B} < 0$ or $\hat{R} < 0$. Here $-\infty$ denotes a “large magnitude” negative number.

(A) Write down a recurrence for $\text{payoff}(\hat{B}, \hat{R}, j)$

(Hint: Consider two cases, either we choose to invest in investment $j$ or not. We will get two recursive calls to $\text{payoff}$ involving $j - 1$, and need to take the maximum between these two values.).
(B) From the recurrence above, write a python function for \( \text{PAYOFF} \). **Do not memoize yet.**

Solution.

```python
def payoff(B, R, j):
    if (B < 0 or R < 0):
        return -Inf
    if (j == 0):
        return 0
    # Choose to invest in j
    p1 = payoff[j] + payoff(B - cost[j], R - risk[j], j-1)
    # Choose not to invest in j
    p2 = payoff(B, R, j-1)
    return max(p1, p2)
```

(C) What would be the arguments to the \( \text{PAYOFF} \) function above to compute the maximum payoff considering all \( n \) investments, a budget of \( B \) and maximum possible risk of \( R \)?

Solution.

\( \text{PAYOFF}(B, R, n) \)

(D) Write down the Python code (now use memoization) to compute the \( \text{PAYOFF} \) function and report the value of maximum payoff that it finds for the following set of investments:

<table>
<thead>
<tr>
<th>ID</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost</td>
<td>20</td>
<td>10</td>
<td>40</td>
<td>50</td>
<td>50</td>
<td>70</td>
<td>80</td>
</tr>
<tr>
<td>Payoff</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>10</td>
<td>20</td>
<td>25</td>
<td>35</td>
</tr>
<tr>
<td>Risk Score</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

(D.1) What is the maximum payoff possible with a budget of at most \( B = 120 \) and risk limit \( R = 9 \)? What are the corresponding investments made?

(D.2) What is the maximum payoff possible with a budget of at most \( B = 140 \) and risk limit \( R = 12 \)? What are the corresponding investments?
### Solution.
see invest.py file

**P2 (20 points)** A video game cat is sitting at $x = 0$ and wishes to end up at $x = 101$ in the smallest possible number of leaps. At each stage, the cat can leap forward 1, 2, 3 or 4 steps. Furthermore, there are the following constraints:

1. The positions $x = 1, 4, 9, 16, 25, 36, 49, 64, 81$ and 100 (perfect squares) are electrified. If the cat makes a leap starting from any of these positions, it automatically loses a life.

2. If the cat starts a leap at $x = 42$, it automatically gains an extra life.

3. Initially, the cat has 9 lives (of course), and the game immediately ends in a loss if the cat has lost all its lives and has not reached $x = 101$. However, if the cat reaches $x = 101$, with $l \geq 0$ lives, it wins.

What is the minimum number of hops the cat needs to complete it task? You are asked to derive a dynamic programming solution.

Let $\text{NUMHOPS}(x, l)$ denote the minimum number of hops needed to reach, starting from position $x$ to 101 with $l$ lives. One of the base cases is that $\text{NUMHOPS}(x, 0) = \infty$ for $x < 101$.

(A) Write bases cases for $\text{NUMHOPS}(101, l)$ for $l \geq 0$.

**Solution.** $\text{NUMHOPS}(101, l) = 0.$

(B) Write down a recurrence for $\text{NUMHOPS}(x, l)$ for $x < 101$ and $l > 0$. Your recurrence will need to handle special cases when $x = 1, 4, 9, \ldots, 100$ and $x = 42$.

**Solution.**

\[
\text{NUMHOPS}(x, l) = \begin{cases} 
\min(\text{NUMHOPS}(x + 1, l), \ldots, \text{NUMHOPS}(x + 4, l)) & \text{if } x \neq 1, 4, 9, \ldots, 100, 42 \\
\min(\text{NUMHOPS}(x + 1, l + 1), \ldots, \text{NUMHOPS}(x + 4, l + 1)) & \text{if } x = 42 \\
\min(\text{NUMHOPS}(x + 1, l - 1), \ldots, \text{NUMHOPS}(x + 4, l - 1)) & \text{if } x = 1, 4, 9, \ldots, 100 \\
\infty & x > 101 
\end{cases}
\]

(C) Write a memoized routine in Python and use it to compute $\text{NUMHOPS}(0, 9)$. Augment your routine to also compute the set of hops that the cat needs to make and report that, as well.