Flow Trader

2020 Mar 05 - Hackers rank (7 multiple choices + 3 easy level + 2 median level)
2020 Mar 09 - HR video interview
2020 Mar 12 - Hackers rank : codepair

Hackers rank – Median level questions

Q1. Number of distinct subsequence

Jamie is walking along a number line that starts at point *0* and ends at point *N*. She can move either one step to the left or one step to the right of current location bounded within range *[0,N]*. Given string `str` of movement instruction consisting of the letters `'l'` and `'r'` (denoting the instruction to move one step left and one step right respectively), Jamie has to find out the number of substring of `str` so that by following instructions in the substring she can move from point `x` to point `y`. For example, given `str = 'rrlrlr'` while `x = 1` and `y = 2`, then `ans = 7` (number of substrings), here is the reason :

We need to construct paths that generate net movement one step to right, here are the possibilities …

- `substr = r` (this `substr` can be created by `str[0]`, `str[1]`, `str[3]` or `str[5]`)
- `substr = rrl` (this `substr` can be created by `str[012]`, `str[014]`, `str[034]` or `str[134]`)
- `substr = rlr` (this `substr` can be created by `str[023]`, `str[025]`, `str[045]` , `str[123]` , `str[125]` , `str[145]` or `str[345]`)
- `substr = lrr` (this `substr` can be created by `str[235]` only)
- `substr = rrllr` (this `substr` can be created by `str[01245]` only)
- `substr = rrlrl` (this `substr` can be created by `str[01234]` only)
- `substr = rlrlr` (this `substr` can be created by `str[02345]` or `str[12345]`)

→
```
num_distinct_subsequence("rrlrlr", 1, 2, 6)  = 7
num_distinct_subsequence("rrlrlr", 1, 2, 6) != 20 = 4 + 4 + 7 + 1 + 1 + 1 + 2
```

Solution

The following is brute force search for all distinct substrings, then filter those that starts from `x` and reach `y` without going outside the two bounds. Finding all substrings first can avoid duplicated `is_reachable` invocation on the same substring.

```
def is_reachable(s, x, y, N) :
    pos = x
    for c in s :
        if c == 'l' :
            pos = pos-1
            if pos < 0 : return False
        else :
            pos = pos+1
            if pos > N : return False
    if pos == y : return True
    else : return False

def find_substr(command, substrs) :
    if len(command) == 0 : return

    temp = set()
    find_substr(command[1:], temp)

    for s in temp :
        substrs.add(s)
        substrs.add(command[0]+s)
    substrs.add(command[0:1]) # Dont forget this

def num_walk_along_the_integer_axis(command, x, y, N) :
    substrs = set()
    find_substr(command, substrs)

    count = 0
    for s in substrs :
        if is_reachable(s, x, y, N) :
            count = count + 1
    return count
```

Q2. Checking a point inside a triangle

Write a function to check if a point lies within a triangle defined by 3 two-dimensional points.

Solution

The answer from `area` can be positive or negative, its sign is reversed by changing clockwise point order to anti-clockwise. Thus we have :

```
area(A,B,C) == area(P,B,C) + area(A,P,C) + area(A,B,P)          for all point P and triangle ABC
area(A,B,C) == fabs(area(P,B,C)) + fabs(area(A,P,C)) + fabs(area(A,B,P))   for point P inside triangle ABC
area(A,B,C) != fabs(area(P,B,C)) + fabs(area(A,P,C)) + fabs(area(A,B,P))   for point P outside triangle ABC
```

```
double area(coord A, coord B, coord C)
{
    return (A.x*(B.y-C.y) + B.x*(C.y-A.y)+ C.x*(A.y-B.y)) / 2;
}

bool is_inside(coord A, coord B, coord C, coord P)
{
    auto area0 = area(A,B,C);
    auto area1 = area(P,B,C);
    auto area2 = area(A,P,C);
    auto area3 = area(A,B,P);
    return area0 * (fabs(area1) + fabs(area2) + fabs(area3)) > 0; // all 4 areas are in same sign if P lies inside ABC
}
```

Preparation for HR

Why Flow Trader?

- top tier fin-tech company, elite team, cutting edge tech, international exposure, global mindset, diversified values
- pool of talent, best people in fin/eco/cs/maths, keep learning and moving forward
- create direct impact on firm revenue, source of job satisfaction, driving force to deliver the best
- smile on faces, pursue for smart way of doing thing, making contribution as well

What makes you unique?

- diversified background (from manufacturing to finance, from algo trading to financial engineering)
- specialized in C++ programming, low latency, as well as algo, quantitative skills
- hands on experience, non-technical skills, managing / reporting / presentation

What are your strength?

- hard working, concentrate, humble, eager to learn, detail minded
- responsible, self-discipline, autonomously, take ownership of my work (test + doc)
- dedication to job, loyal to company (seldom change job), efficiency improves over time

What are your weaknesses?

- conservative on risk exposure (for example, a lot effort in testing, exchange simulator)
- theoretical, digging detail (tradeoff perfection and time-to-market, constraint in reality, breakdown into phases)

Describe a challenging project in the past. How did you overcome it?

- one-man band, greenfield project start from scratch
- self-discipline, autonomy with not much guidance
- self-learning, continuous improvement
- persistence and perseverance, strive for excellence, push beyond limit

Do you have any questions?

- What are day-to-day responsibilities of the role?
- What are you looking for in the candidate?
- May I know a little bit about culture and environment?

Hackers rank : codepair

Implement printer with variadic template

```
template<typename ARG> // Don't forget this boundary case
printer() {}

template<typename ARG>
printer(const ARG& arg) { std::cout << arg; }

template<typename ARG, typename... ARGS>
printer(const ARG& arg, ARGS&&... args)
{
    printer(arg);
    printer(args...);
}
```

Implement classify traits for int, double

```
template<typename T> struct is_int : std::false_type {};
template<> struct is_int<int> : std:: true_type {};
```

Implement classify traits for int, double and others

```
template<typename T> void f(const T& x)
{
      if (std::is_integral<T>::value)                        std::cout << "This is an integer.";
      else if (std::is_floating_point<T>::value)             std::cout << "This is a floating point.";
      else                                                   std::cout << "This is something else.";
}
```

How to make the above decision done in compile time?

```
template<typename T> void f(const T& x)
{
      if constexpr(std::is_integral<T>::value)               std::cout << "This is an integer.";
      else if constexpr(std::is_floating_point<T>::value)    std::cout << "This is a floating point.";
      else                                                   std::cout << "This is something else.";
}
```

Application of `if constexpr` in template programming

```
template<typename T> auto length(const T& value) noexcept
{
      if constexpr (std::is_integral<T>::value) return value;
      else return value.length();
}
auto x = length(123);
auto y = length(std::string{abcdef});
```

Check if a string is sorted according to a dictionary (which is also a string)

```
bool is_sorted_according_to_dict(const std::string& s, const std::string& dict)
{
      int n=0;
      for(auto& c:dict)
      {
            if (s[n] == x) ++n;
            if (n == s.size()) return true;
      }
      return false;
}
```

Extend this to generic objects :

```
template<typename ITER0, typename ITER1>
bool is_sorted_according_to_dict (    const ITER0& s_begin, const ITER0& s_end,
                                      const ITER1& d_begin, const ITER1& d_end)
{
      auto s_iter = s_begin;
      for(auto d_iter = d_begin; d_iter!=d_end(); ++d_iter)
      {
            if (*s_iter == *d_iter) ++s_iter;
            if ( s_iter == s_end) return true;
      }
      return false;
}
```