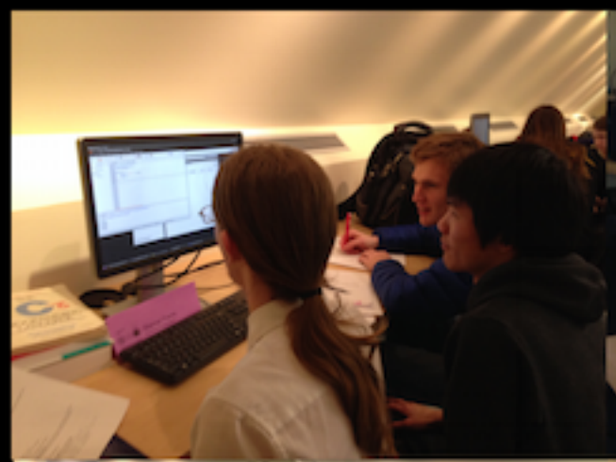
ACM-ICPC and you

Daniel Epstein, CSE Grad Student

Shameless plug #1

I’m organizing this year’s qualifying contest




acm
International Collegiate Programming Contest
Pacific NW Region
Puget Sound Site Champions
Division I
November 15, 2014

Shameless plug #2

Saturday, October 24, 9:00am-4pm, CSE Labs 002 & 006
(tell your friends)

Reasons why you should come

- Free food
- Hang out and code with friends
- Great prizes from Google
- Practice problem-solving skills
- You get to write bad code and nobody will yell at you

What are programming competitions?

3 people!

1 computer!

6512 Assignments

When Starfleet headquarters gets a request for an exploration expedition, they need to determine which ship from those currently docked in the docking bay to send. They decide to send whichever ship is currently able to make the expedition based on how much fuel is currently stored on the ship as well as how long it will take the ship to arrive at the expected destination. Due to the age and current maintenance of the ships, each ship travels at a different top speed and has a different fuel consumption rate. Each ship reaches its top speed instantaneously.

Input

Input begins with a line with one integer T ($1 \leq T \leq 50$) denoting the number of test cases. Each test case begins with a line with two space-separated integers N and D, where N ($1 \leq N \leq 100$) denotes the number of ships in the docking bay and D ($1 \leq D \leq 10^6$) denotes the distance in light-years to the expedition site. Next follow N lines with three space-separated integers v_i, f_i, and c_i, where v_i ($1 \leq v_i \leq 1000$) denotes the top speed of ship i in light-years per hour, f_i ($1 \leq f_i \leq 1000$) denotes the fuel on ship i in kilos of deuterium, and c_i ($1 \leq c_i \leq 1000$) denotes the fuel consumption of ship i in kilos of deuterium per hour.

Output

For each test case, print a single integer on its own line denoting the number of ships capable of reaching the expedition site. Be careful with integer division!

Sample Input

```
2
3 100
52 75 10
88 13 44
56 9 5
2 920368
950 950 1
943 976 1
```

Sample Output

```
2
1
```

6512 Assignments

When Starfleet headquarters gets a request for an exploration expedition, they need to determine which ship from those currently docked in the docking bay to send. They decide to send whichever ship is currently able to make the expedition based on how much fuel is currently stored on the ship as well as how long it will take the ship to arrive at the expected destination. Due to the age and current maintenance of the ships, each ship travels at a different top speed and has a different fuel consumption rate. Each ship reaches its top speed instantaneously.

Input

Input begins with a line with one integer T ($1 \leq T \leq 50$) denoting the number of test cases. Each test case begins with a line with two space-separated integers N and D, where N ($1 \leq N \leq 100$) denotes the number of ships in the docking bay and D ($1 \leq D \leq 10^6$) denotes the distance in light-years to the expedition site. Next follow N lines with three space-separated integers v_i, f_i, and c_i, where v_i ($1 \leq v_i \leq 1000$) denotes the top speed of ship i in light-years per hour, f_i ($1 \leq f_i \leq 1000$) denotes the fuel on ship i in kilos of deuterium, and c_i ($1 \leq c_i \leq 1000$) denotes the fuel consumption of ship i in kilos of deuterium per hour.

Output

For each test case, print a single integer on its own line denoting the number of ships capable of reaching the expedition site. Be careful with integer division!

Sample Input

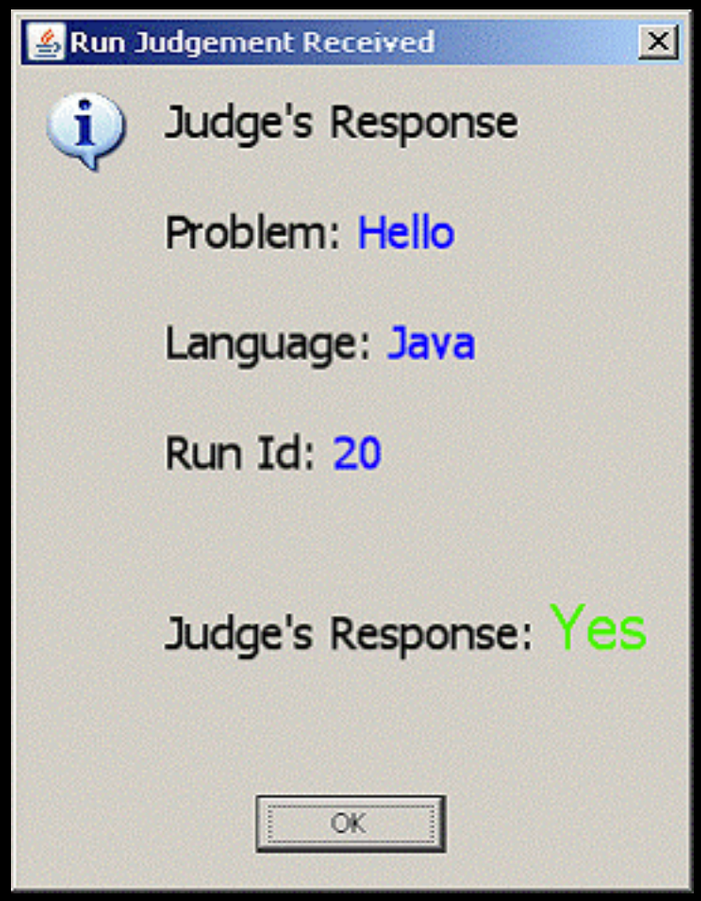
```
2
3 100
52 75 10
88 13 44
56 9 5
2 920368
950 950 1
943 976 1
```

Sample Output

```
2
1
```

```java
import java.util.*;
import java.io.*;

public class Main {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    int cases = in.nextInt();
    for(int i=0;i<cases;i++) {
      int n = in.nextInt();
      int d = in.nextInt();
      int count = 0;
      for(int j=0;j<n;j++) {
        int v = in.nextInt();
        int f = in.nextInt();
        int c = in.nextInt();
        if((v*f)/c >= d) {
          count++;
        }
      }
      System.out.println(count);
    }
  }
}
```


Run Judgement Received
Judge's Response
Problem: Hello
Language: Java
Run Id: 20
Judge's Response: Yes
OK

No - Compilation Error

No - Runtime Exception

No - Time Limit Exceeded

No - Wrong Answer

No - See Contest Staff

No - Compilation Error

No - Runtime Exception

No - Time Limit Exceeded

~10,000,000 operations

No - Wrong Answer

No - See Contest Staff

C

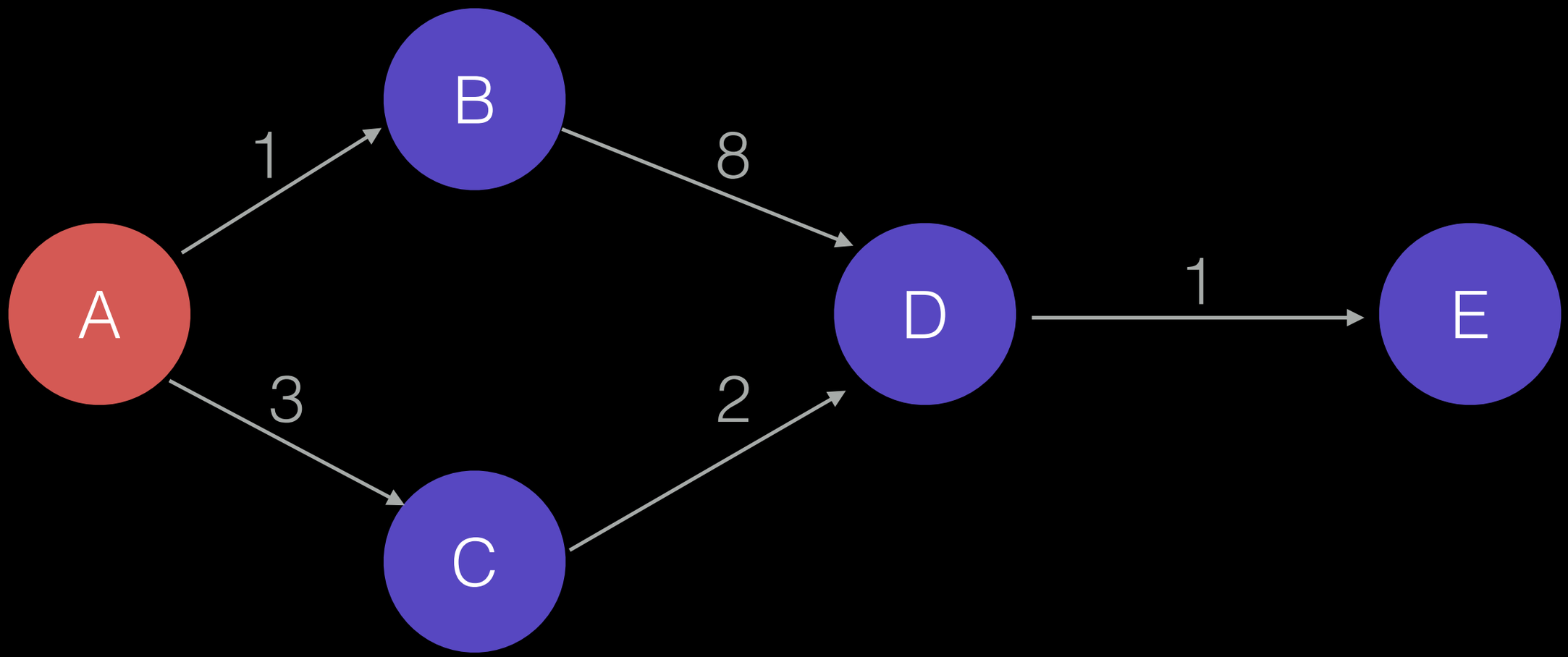
C++

Java

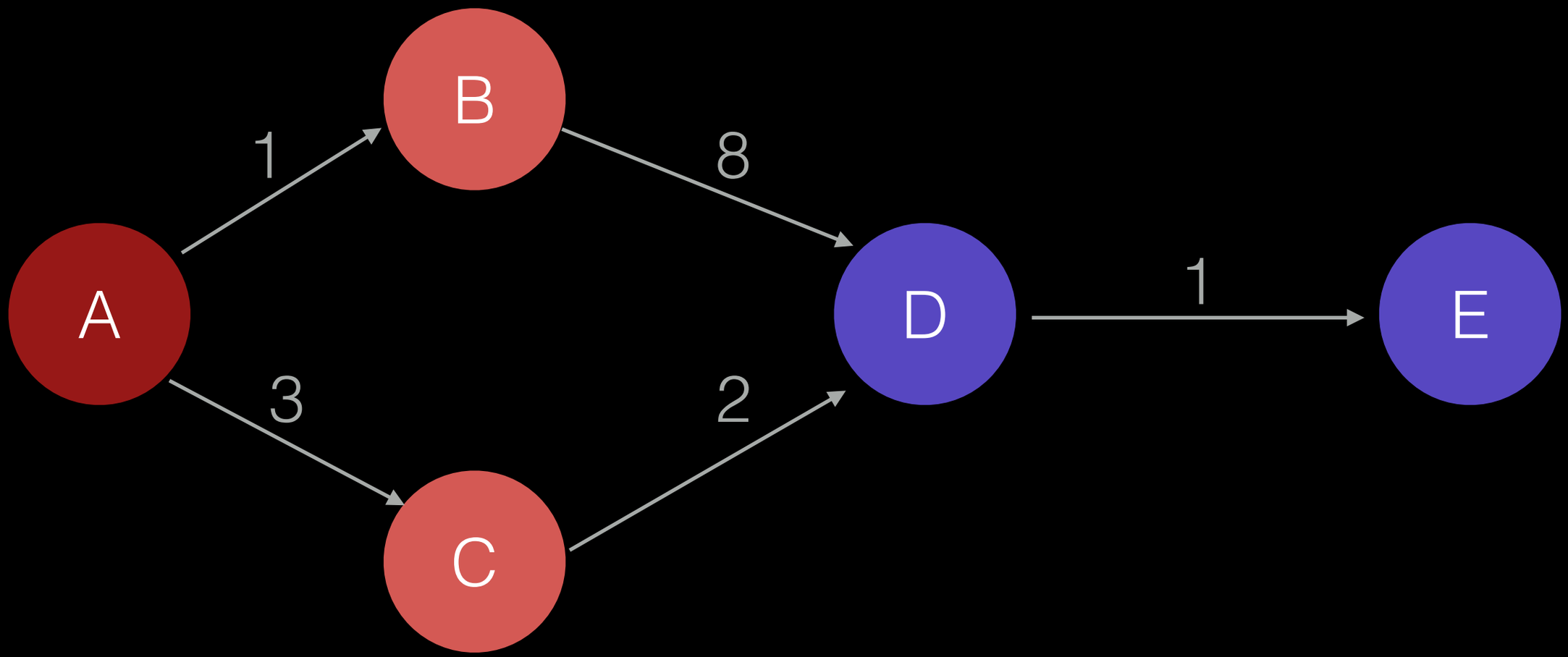
Python*

Shameless Plug #3

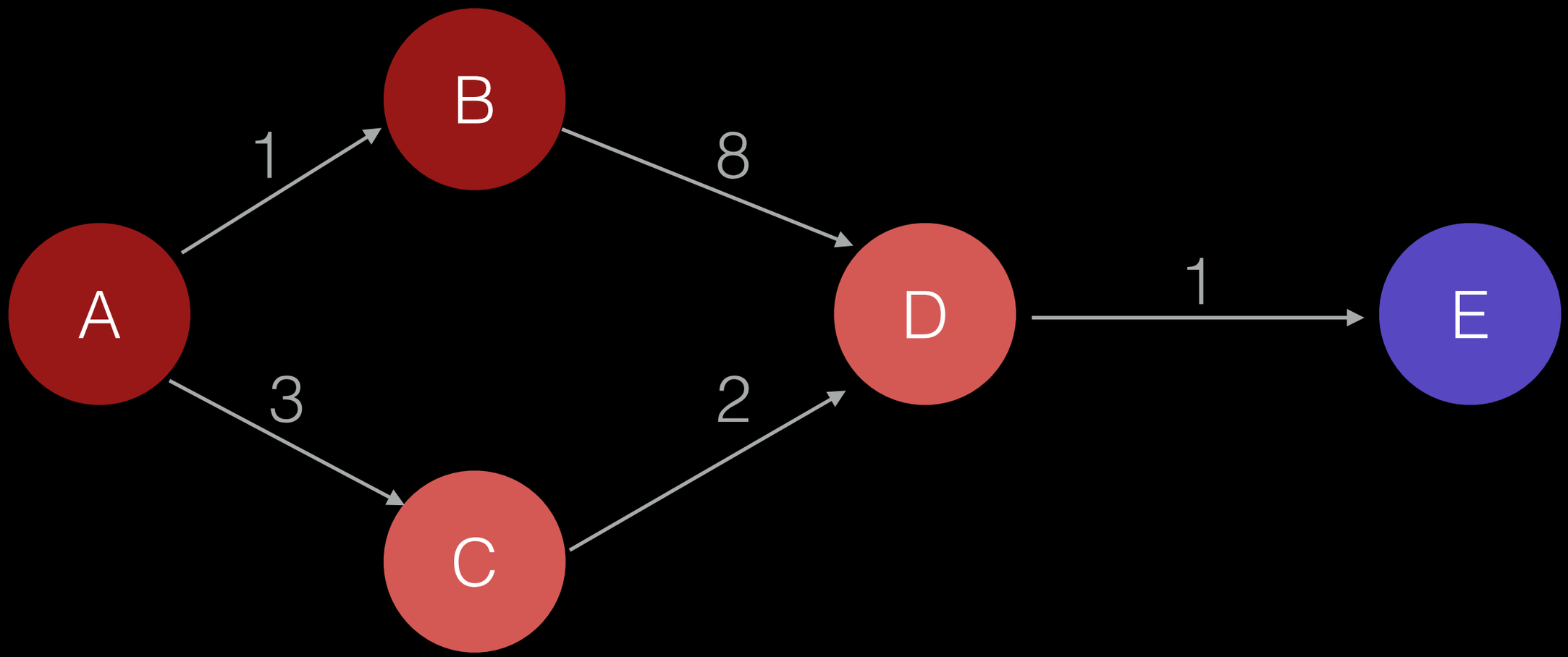
https://github.com/depstein/programming-competitions



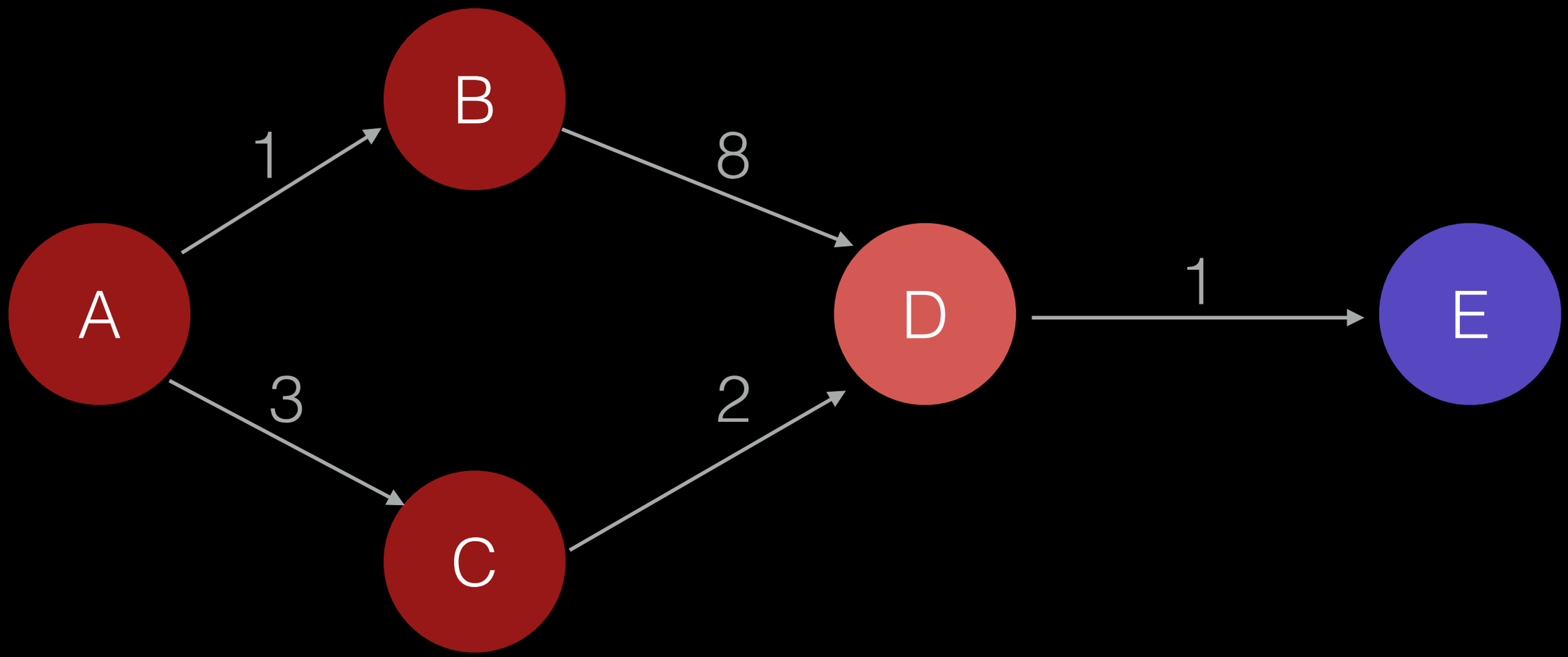
A 0



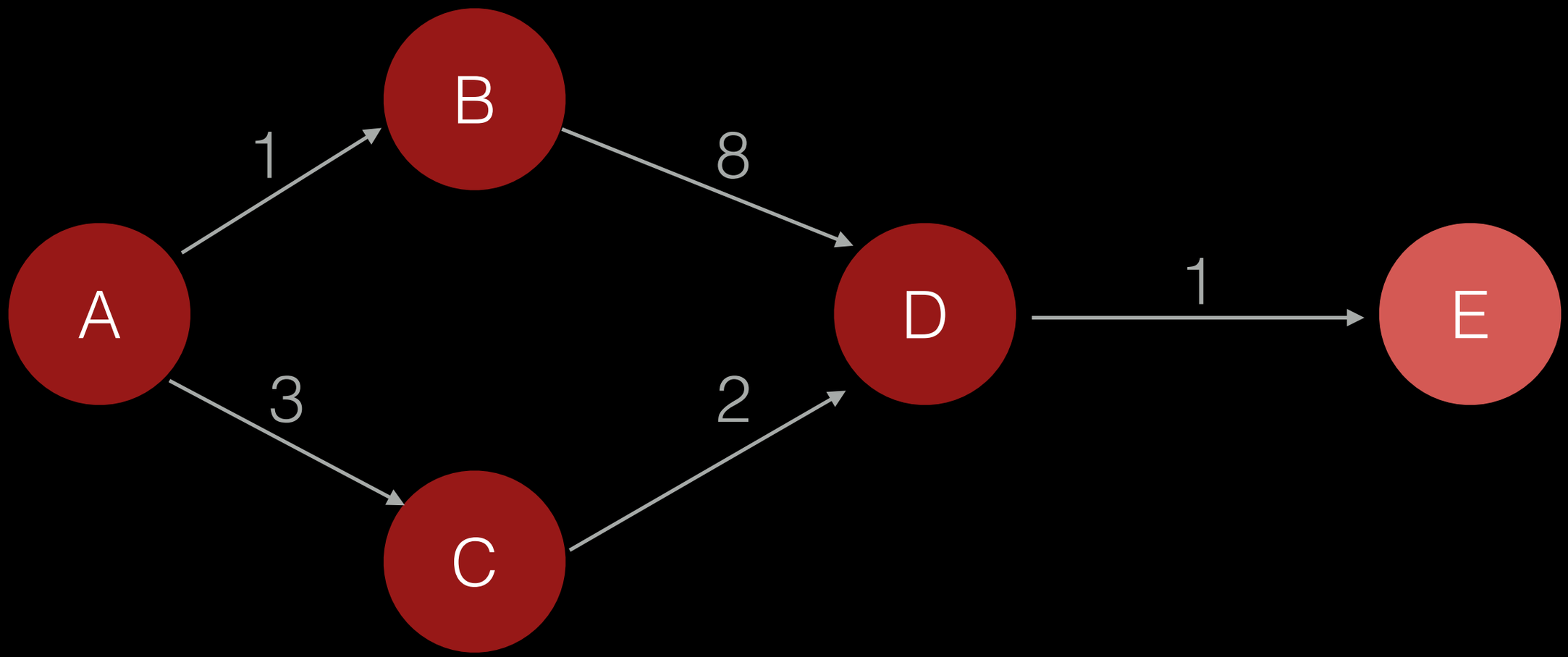
A	B	C
0	1	3



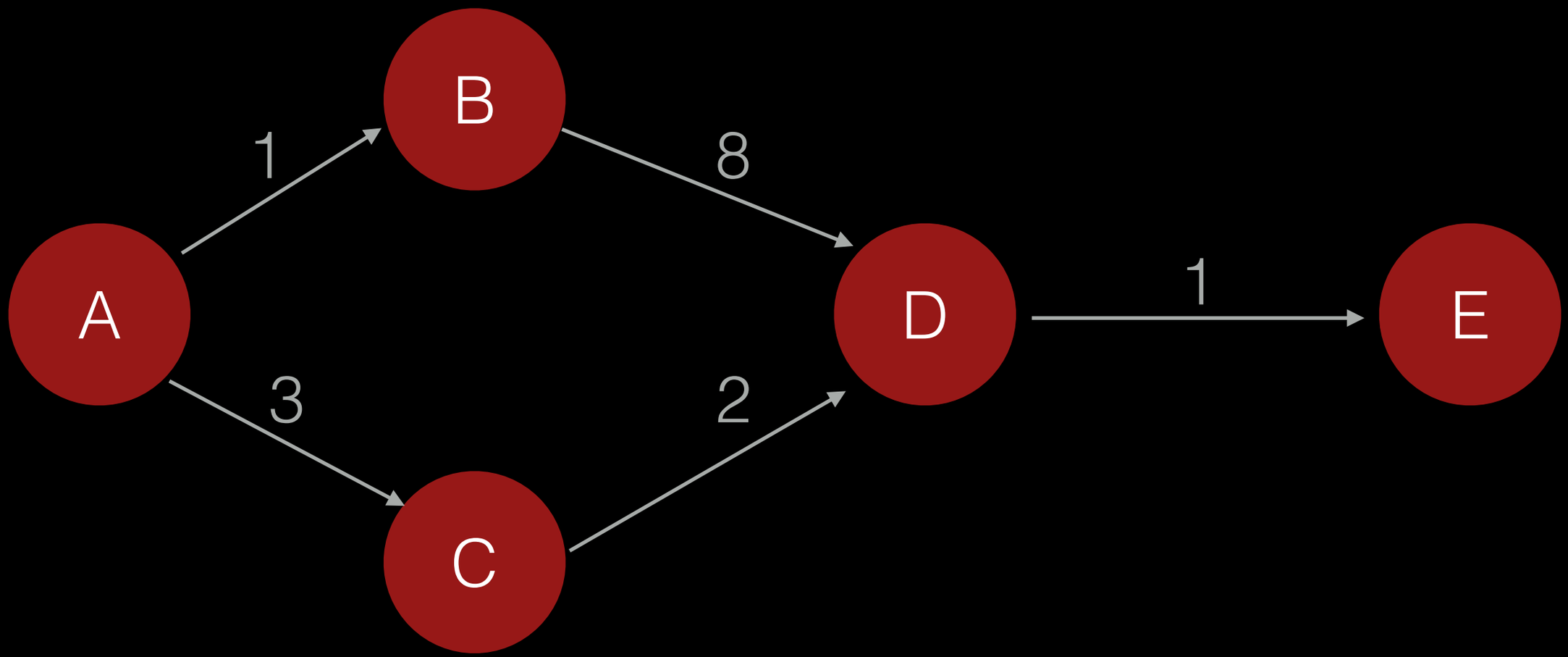
A	B	C	D
0	1	3	9



A	B	C	D
0	1	3	5



A	B	C	D	E
0	1	3	5	6



A	B	C	D	E
0	1	3	5	6

```java
public static void dijkstra(Node root, ArrayList<Node> allNodes) {
  PriorityQueue<Node> q = new PriorityQueue<Node>();
  root.distance = 0;
  q.add(root);

  while(q.size() > 0) {
    Node u = q.poll();
    for(Node n : u.edges.keySet()) {
      if(n.distance == Integer.MAX_VALUE) { // Update the distance to node n
        q.remove(n);
      }
      n.distance = Math.min(n.distance, u.distance + u.edges.get(n));
      q.add(n);
    }
  }
}
```

```java
public static void bfs(Node root, ArrayList<Node> allNodes) {
  Queue<Node> q = new LinkedList<Node>();
  root.distance = 0;
  q.add(root);

  while(q.size() > 0) {
    Node u = q.poll();
    for(Node n : u.edges) {
      if(n.distance == Integer.MAX_VALUE) { // Has not been visited yet
        n.distance = u.distance + 1;
        q.add(n);
      }
    }
  }
}
```

```java
public static void dijkstra(Node root, ArrayList<Node> allNodes) {
  PriorityQueue<Node> q = new PriorityQueue<Node>();
  root.distance = 0;
  q.add(root);

  while(q.size() > 0) {
    Node u = q.poll();
    for(Node n : u.edges.keySet()) {
      if(n.distance == Integer.MAX_VALUE) { // Update the distance to node n
        q.remove(n);
      }
      n.distance = Math.min(n.distance, u.distance + u.edges.get(n));
      q.add(n);
    }
  }
}
```

```java
public static void bfs(Node root, ArrayList<Node> allNodes) {
  Queue<Node> q = new LinkedList<Node>();
  root.distance = 0;
  q.add(root);

  while(q.size() > 0) {
    Node u = q.poll();
    for(Node n : u.edges) {
      if(n.distance == Integer.MAX_VALUE) { // Has not been visited yet
        n.distance = u.distance + 1;
        q.add(n);
      }
    }
  }
}
```

```java
public static void dijkstra(Node root, ArrayList<Node> allNodes) {
  PriorityQueue<Node> q = new PriorityQueue<Node>();
  root.distance = 0;
  q.add(root);

  while(q.size() > 0) {
    Node u = q.poll();
    for(Node n : u.edges.keySet()) {
      if(n.distance == Integer.MAX_VALUE) { // Update the distance to node n
        q.remove(n);
      }
      n.distance = Math.min(n.distance, u.distance + u.edges.get(n));
      q.add(n);
    }
  }
}
```

More about the contest

	Problem A	Problem B	Problem C	Problem D	Total
Team 1	—	—	—	—	0 solved 0 mins
Team 2	—	—	—	—	0 solved 0 mins
Team 3	—	—	—	—	0 solved 0 mins

	Problem A	Problem B	Problem C	Problem D	Total
Team 3	—	—	Solved! 1 try 5 mins	—	1 solved 5 mins
Team 1	—	—	—	—	0 solved 0 mins
Team 2	—	—	—	—	0 solved 0 mins

	Problem A	Problem B	Problem C	Problem D	Total
Team 3	—	—	Solved! 1 try 5 mins	—	1 solved 5 mins
Team 2	Solved! 1 try 8 mins	—	—	—	1 solved 8 mins
Team 1	—	—	—	—	0 solved 0 mins

	Problem A	Problem B	Problem C	Problem D	Total
Team 3	—	—	Solved! 1 try 5 mins	—	1 solved 5 mins
Team 2	Solved! 1 try 8 mins	—	—	—	1 solved 8 mins
Team 1	—	—	Unsolved 1 try	—	0 solved 0 mins

	Problem A	Problem B	Problem C	Problem D	Total
Team 3	—	—	Solved! 1 try 5 mins	—	1 solved 5 mins
Team 2	Solved! 1 try 8 mins	—	—	—	1 solved 8 mins
Team 1	Solved! 1 try 14 mins	—	Unsolved 1 try	—	1 solved 14 mins

	Problem A	Problem B	Problem C	Problem D	Total
Team 1	Solved! 1 try 14 mins	—	Solved! 2 tries 18 mins	—	2 solved 14+18+20= 52 mins
Team 3	—	—	Solved! 1 try 5 mins	—	1 solved 5 mins
Team 2	Solved! 1 try 8 mins	—	—	—	1 solved 8 mins

Solving problems is good!

Guessing is bad, but only if you eventually figure out the solution

acm International Collegiate Programming Contest

IBM. event sponsor

Pacific NW Region

Host a Contest! | Contest Rules | Registration | Contest Details | Results | Links

Welcome to the Pacific NW Region Programming Contest! The Pacific NW Region is comprised of the following areas: Alaska, Hawaii, British Columbia, Washington, Oregon, northern/central California and western Nevada. Because of the large geographic area of the region, the contest is held simultaneously at multiple sites: Northern California, Northwest (Oregon), Northeast (E. WA and Idaho), Puget Sound (Western Washington), Canada, and Hawaii.

Announcements

- **UPDATED 10/5: REGISTRATION IS NOW OPEN! THANK YOU FOR YOUR PATIENCE!**
- **2015 Contest will be Saturday, November 14**
    - Confirmed sites: Canada - Simon Fraser, Oregon - George Fox, Hawaii - Brigham Young Hawaii, Northeast - Eastern Washington University, Puget Sound/ Western Washington - University of Puget Sound, Northern California - UC Berkeley
- There are two divisions in 2015 (just as in 2014) **but with some changes to D2**
    - Division 1 (D1) is for teams that are very strong algorithmically. The D1 problem set will be difficult. It will be along the lines of a lite version of what you would see at World Finals. Only D1 teams are eligible for slots in the World Finals.
    - Division II (D2) is approachable for teams that have not completed the algorithm sequence at their school and/or have not competed in programming contests before. Almost all problems in D2 can be solved using techniques at the CS1 and CS2 levels. Most teams in this division have a great chance at solving many problems. If a team does well in this division this year, that team should try D1 next year! Participating in D2 will not count against your eligibility for World Finals in future years (other than the rules established by ICPC itself based on age/year in school -- see the Contest Rules tab above for more information on the ICPC rules). When registering a team for D2, include 'D2' at the beginning of the team name.
    - The following zip file contains three problems that serve as a baseline for competing in Division 1. If a team cannot solve these three problems, that team should compete in Divsion 2. The file contains input, output, and solutions to each problem along with a README that includes hints. The file size is ~15MB.
        - Div1BaselineProblems.zip
    - **Check the two division FAQ for more details**
- We will once again support Python. While Python is not yet supported in World Finals, it looks like it may be supported starting in 2017. We will support Python 2.7.6 and 3.4.0. We are also adding support for C# via the Mono Project.
- **IMPORTANT NOTE**: In the future, if you are interested and capable of hosting, please contact the regional director Tom Capaul at: tcapaul@ewu.edu and/or director@acmicpc-pacnw.org. Costs of the contest (food, t-shirts, balloons, etc.) are covered by the PacNW region. As a host school your teams participate for free.

ACM-ICPC World Finals

May 15 - 20

2016

Phuket, Thailand

host Prince of Songkla University

IBM event sponsor

world finals
- Schedule
- Activities
- Local Information
- Teams
- World Finals Rules
- On-Site Registration
- Video/Photo Coverage
- World Finals Results
- Past Problems
- Fact Sheet
- Prog. Environment

regionals
- Regional Finder
- Upcoming Regionals
- Regional Results
- Regional Rules
- Getting Involved
- Starting a Regional
- Free ACM Membership

compete
- Preparation
- Policies & Procedures
- FAQs
- The Problems

community
- IBM
- Upsilon Pi Epsilon
- ACM
- Fact Sheet
- ICPC Tools
- History
- Contacts

Participate in Regionals Now!

Find a regional contest

I am a
Coach Contestant Volunteer

Think. Create. Solve.

ICPCNews

Connect.
(and click for more!)

Share your story.
#ICPC2015

acm icpc

IBM event sponsor

international collegiate programming contest

world map | what's new | about icpc

ICPCNews 3h
@ICPCNews

#UVa Online Judges Automatic Teaching Presenters in Valladolid, Spain!
buendia.uva.es/jueces-automat… … #inspiring
pic.twitter.com/Q5iwi1LTdS

Expand

ICPCNews 23h
@ICPCNews

Happy Friday! Watch the ICPCAnalytics video

Important Times

- Monday, October 19, 5pm: final deadline to register a team
- Saturday, October 24, 9am in EEB-105: contest starts

Questions?

Daniel Epstein
depstein@cs.washington.edu
https://github.com/depstein/programming-competitions