Sponsored by FIELD AGENT

2019 JBU Programming Competition

Date: **Saturday, February 2nd, 9:00am–5:30pm**

Location: **Balzer Technology Center**

Cost: **FREE (including lunch and dinner*)**

Registration: **February 1st Deadline (team of 3 or 4 students)**

Schedule: **Check-In and Practice Contest – 9:00am to 9:30am**
Contest – 9:30am to 5:00pm
BTC Tour – 5:00pm
Awards Ceremony – 5:15pm, in BTC 221
*Optional – Free Dinner for high school students
– 5:30pm, in Kresge Dining Hall

Details: http://www.jbu.edu/majors/engineering/events/

Sponsored by FIELD AGENT

Prizes

2019 JBU Computer Programming Competition

Saturday, February 2nd, Balzer Technology Center

1st Place **$20 per team member**

AND

$1000 JBU Scholarship *

2nd Place **$10 per team member**

AND

$500 JBU Scholarship *

3rd Place **$5 per team member**

AND

$250 JBU Scholarship *

*JBU scholarships are for each high school contestant on the winning team. This scholarship cannot be used to create a combined total scholarship exceeding $16,000, when combined with any other JBU scholarships or remission.

John Brown University Computer Programming Contest Rules

1. No food or drink in the computer rooms!

2. Contestants will work solutions to the problems using C, C++, C#, Java, or Python3. All solutions must read from standard input and write its output to standard output, using the DomJudge web interface http://domjudge.jbu.edu/domjudge/team/ , with manual available here: https://www.domjudge.org/docs/team-manual.pdf

3. Internet access will not be allowed, except for reference materials (not forums) on the following websites: docs.python.org , docs.oracle.com , msdn.microsoft.com , docs.microsoft.com , www.cplusplus.com , and www.wikipedia.org . You are not allowed to use any electronic form of help or documentation during the contest (other than that provided from within the IDE itself, such as Visual Studio, NetBeans, or PyCharm), including smartphones, flash drives, CDs/DVDs, files from the network or internet, or other programmable devices. You may bring and use any books, notes, or papers that you think may be useful. Textbooks on flash drives will not be permitted. You may bring printed copies of any programs that you would like.

4. In the event that you feel a problem statement is ambiguous, you may request a clarification. Read the problem carefully before requesting a clarification. If the judges do not believe that you have discovered an ambiguity in the problem, the judges will respond that no clarification is necessary. If you receive this response, you should read the problem description more carefully. If you still feel there is an ambiguity, you will have to be more specific or descriptive of the ambiguity you have found. If the problem statement is ambiguous in specifying the correct output for particular input, please include that input data in the clarification request.

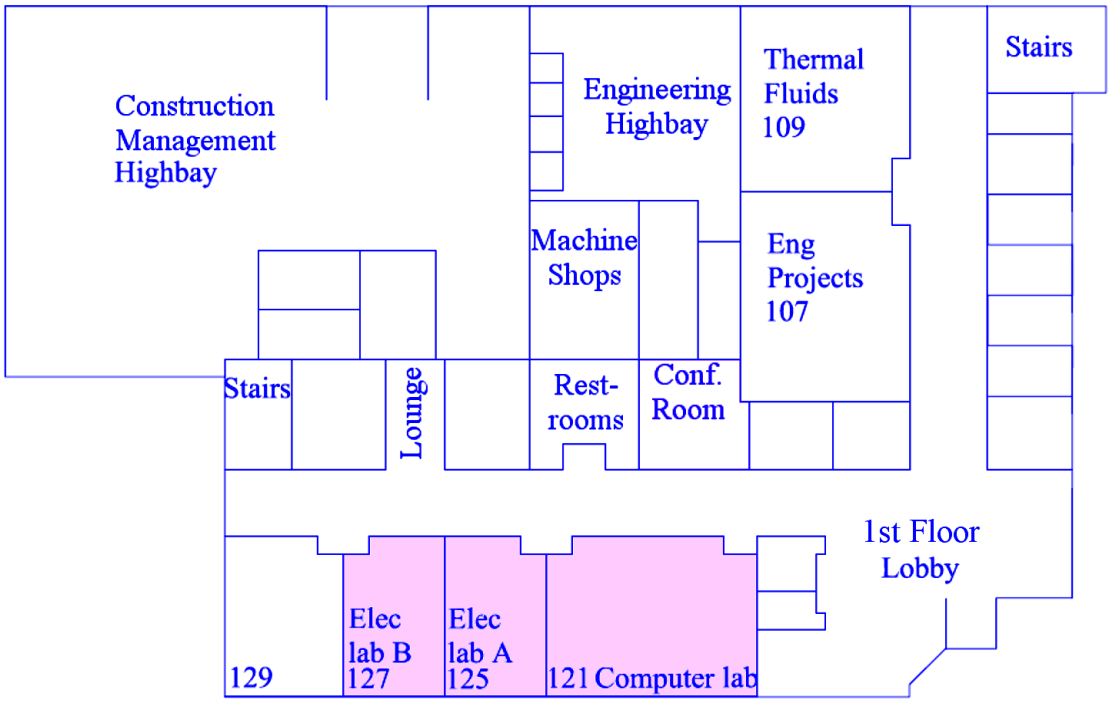
5. Contest judging in the DomJudge system is based upon 3 components: the number of correct problems submitted, the elapsed time from the beginning of the contest to when problems are correctly submitted, and the number of incorrect submissions for problems for which a correct solution is eventually submitted. Teams are first ranked by # of problems for which a correct solution is submitted. In the event two or more teams solve the same number of problems, penalty minutes will be used as a tiebreaker. The winner will be the team with the least penalty minutes. Penalty minutes are calculated using the following:
   a. For each solved problem, the number of minutes from the beginning of the contest until the correct solution was submitted.
   b. For each problem which is eventually solved, a 15 minutes penalty will be accessed for each incorrect solution submitted prior to the correct solution. No penalty will be incurred for incorrect solutions for problems for which a correct solution is never submitted.

In the unlikely event there is still a tie after using penalty minutes, the winner will be the team which achieved its score first.

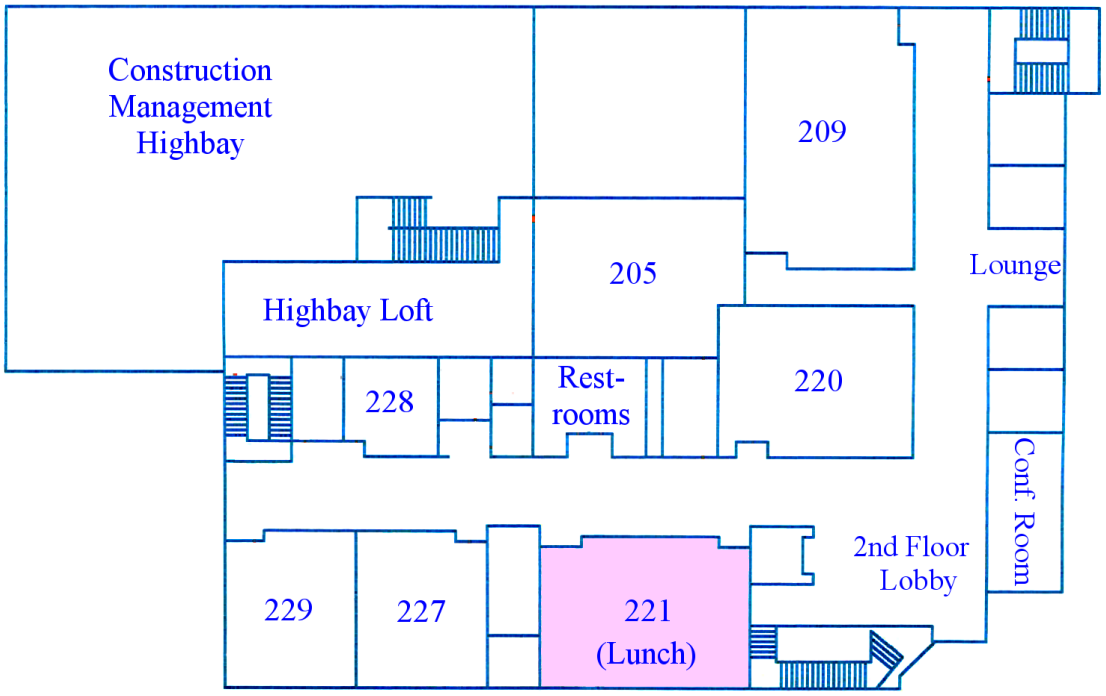
6. Judges' decisions will be final. Judges have the right to amend these rules if necessary during the course of the competition. Judges have the right to disqualify teams for unprofessional conduct.

7. HAVE FUN!

Balzer Technology Center 1st Floor



Balzer Technology Center 2nd Floor



Shaded areas indicate location of the JBU Computer Programming Competition.
Lunch will be served upstairs in room 221. Food and drink are not allowed in the computer labs. Places to sit and eat include BTC221, the Lobbies of the 1st and 2nd floor, and the Lounges of the 1st and 2nd floor.

Building Key

Admissions/Financial Aid **29**
Ampitheater **22**
Balzer Technology Center **4**
Bell Science Hall **6**
Berry Performing Arts Center **3**
Bill George Arena **15**
Campus Safety **21**
CARE Clinic **1**
Cathedral of the Ozarks **25**
Central Plant **21**
Chapman Administration Building **9**
Facilities Services Building **2**
Health Education Building **5**
Honors Center **30**
Hutcheson Hall **17**
Information Technology (ITS) **31**
J. Alvin Brown Residence Hall **20**
Kresge Dining Hall **8**
Learning Resource Center **10**
Mabee Center **8**
Mayfield Residence Hall **27**
Murray Sells Gymnasium **12**
Northslope Apartments **19**
Sager Cabin **16**
Simmons Great Hall **7**
Soderquist Business Center **26**
Student Counseling Center **28**
Townhouses **18**
Walker Residence Hall **14**
Walker Student Center **13**
Walton Lifetime Health Complex **11**
Windgate Visual Arts East **24**
Windgate Visual Arts West **23**

JOHN BROWN UNIVERSITY

2000 West University Street, Siloam Springs, AR 72761
1-877-JBU INFO, 479-524-9500, www.jbu.edu

Hints – John Brown University Computer Programming Contest

Programming Contest problems vary in difficulty from easy to challenging, and are not necessarily sorted in order of difficulty. The problems typically have many common characteristics. If you can come to the contest with good skill at solving the routine tasks like I/O, converting between a string/character/integer/etc, or searching through a loop for a particular value, then you can spend your time/effort on the more challenging aspects of the problems. Below are some hints and common tasks that occur on many programming contest problems. All of these tasks may/may not be needed for this particular contest, but they should give you an idea of some things to study up on for the contest.

1. Most problems will require you to read input from standard input, and write your output to standard output. Be sure you have the basics of I/O well understood.
2. You may bring all the written material you want to the contest, including books and program listings (however, textbooks in electronic format on a flash drive or laptop are not allowed). If you do some practice programs (for example, practicing the techniques discussed on this page), feel free to bring the printouts. You will NOT have internet access during the contest, except for possibly a few reference sites.
3. Termination of the input. Some input sets indicate that the input is over by putting a special value on the last line. For example, the first line of each test case may indicate the number of widgets used in that test case. A test case whose number of widgets == 0 indicates the end of the input file. In other inputs, the end of the input is indicated by simply hitting the end of the input. For example, you are asked to read in a series of numbers, on per line, from the input. You don't know how many numbers there are, so you need to look for the end of the stream. You should be able to quickly handle different types of input termination.
4. Number of elements on a line of input. In some inputs, a line of input may be considered a single string. More typically, however, the lines will contain a series of values. In some cases, the problem may tell you specifically that there are 4 items per line, and you can just read in those items. In other cases, one of the first values in the line (or from a preceding line) will tell you how many more values to read on the line. Probably the hardest case is where the number of values on each line is not specified by the problem description, and you need to break up the input into an unknown number of pieces. You should have a basic idea how to do this, and it wouldn't hurt to practice writing a sample program or two.
5. Problems sometimes ask you to print a blank line after every test case except the last. Hint: That is the same as saying print a blank line before every test case except the first. It's a lot easier knowing when you are printing the first test case than to know you are on the last case (since you probably haven't read the input file to see if there are more cases). Likewise, if asked to put a blank space after each value printed on a line, except the last, that is the same as putting a blank space before each value except the first. Regarding output format, many problems ask you to leave a blank line after all test cases, except the last. This is the same as saying, "leave a blank line before all test cases except the first." The same trick applies to "leave a space after every word on the line except the last."

6. You should be able to handle formatting of the output. In particular, know how to print floating point numbers to a particular number of decimal places. For example, if we ask you to print a dollar amount, $13.4520578 is probably not a good dollar amount.

7. You should be comfortable with the relationship between strings and characters. Given a string, you should be able to work with it as if it is an array of characters. You should be able to quickly convert an ASCII value into an integer and an integer into its ASCII value. You should also be familiar with arrays and vectors, and with loops.

8. Make liberal use of `cout` or `print` statements from your code, to debug your programs as you are writing them. In general, the fastest way to successfully code a program is to compile frequently, starting out with "stubs" or comments where the different pieces of your algorithm will go. Then as time goes on, you can test and fill-in the different missing pieces, testing each one with known sample data as you go. Also, if your code is longer than one screen, you will want to use comments to remind yourself about the purpose of various statements.

9. Generally, your problem submissions will be automatically graded in the DomJudge system by comparing to an approved solution. Be very careful with your output format to make it precisely what the problem specifies.

10. The test cases used by the judges will likely be more complicated than the examples shown in the problem. As you make your sample input files to test the outputs of your program, think about the problem and look for special cases. If the problem specifies that a particular input value will be an integer, could it be negative? Could it be zero? If the problem description doesn't specify otherwise, some test case may exercise this. If you have questions during the contest, request a clarification. Use scrap paper to diagram and think through your test cases.

11. Watch the scoreboard. If everyone else is solving a particular problem, maybe your team should think about attacking it.

12. Teamwork is important. Some teams prefer to work together on each problem, whereas in other teams, one person starts coding the basic I/O for the problem while the other person thinks about the algorithm to solve the problem.

13. Feel free to practice by solving other computer programming contest problems posted on the internet. Don't be discouraged if you can't figure out difficult problems from national and world competitions. Start with the simpler ones and work your way up. Some of the JBU problems will be designed to be easy, and others will be more challenging, so that there will be a range.

14. Have fun!!!

(Adapted by Dr. Tim Gilmour from a document originally written by Dr. David Gallagher)