Solution to Homework 1

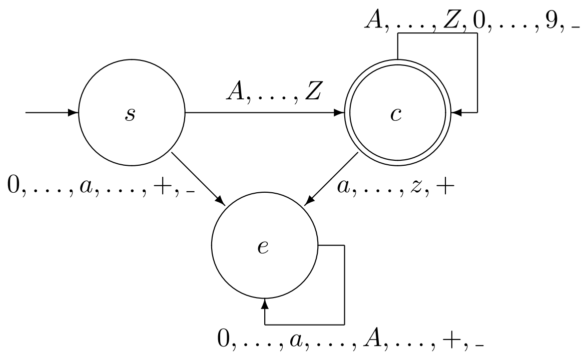
Task 1: general description. In class, we designed automata for recognizing integers and real numbers.

Task 1.1. Use the same ideas to describe an automaton for recognizing names of constants in Java. In Java, this name should start with a letter and consist of all caps; digits and the underscore symbol are also allowed. To describe an automaton, draw a picture like we did in class.

A natural idea is to have 3 states: start (s), correct constant name (c), and error (e). Start is the starting state, c is the only final state. The transitions are as follows:

- from s, any capital letter $A, \ldots, Z$ lead to c, every other symbol leads to e;
- from c, any capital letter, digit, or the underscore symbol lead back to c, every other symbol leads to e;
- from e, every symbol leads back to e.

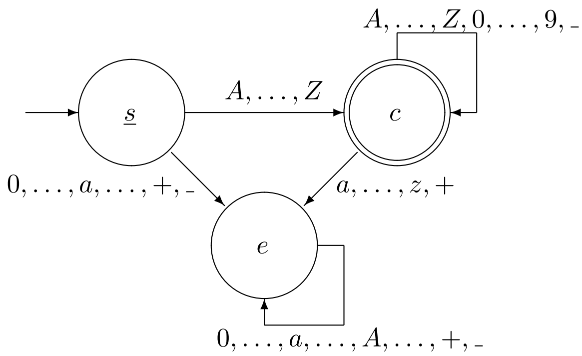
Solution. The desired automaton takes the following form:



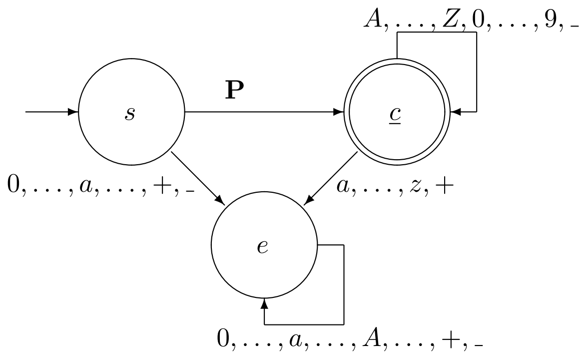
Task 1.2. Trace, step-by-step, how the finite automaton from Part 1.1 will check whether the following two words (sequences of symbols) are correct names for Java constants:

- the word $PI2$ (which this automaton should accept) and
- the word $Pi2$ (which this automaton should reject).

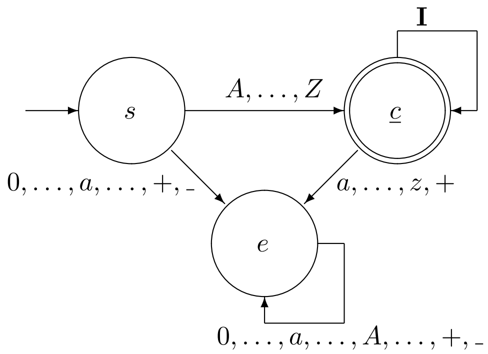
Solution. Let us trace how this automaton will accept the word $PI2$. We are originally in the state s:



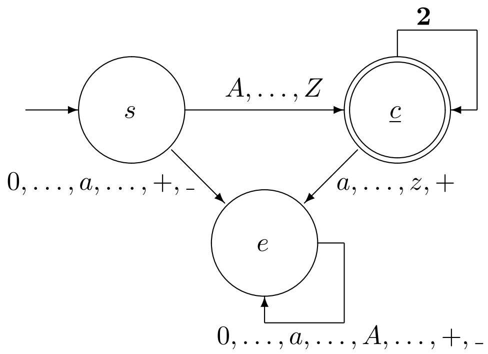
Then, we read the first letter P of the word $\mathbf{P}I2$, so we move to state c:



Then, we read the second letter I of the word $P\mathbf{I}2$, and we stay in the state c:

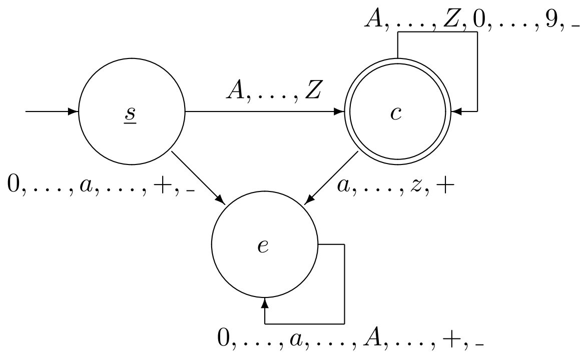


Then, we read the third symbol 2 of the word $PI\mathbf{2}$, and we stay in the state c:

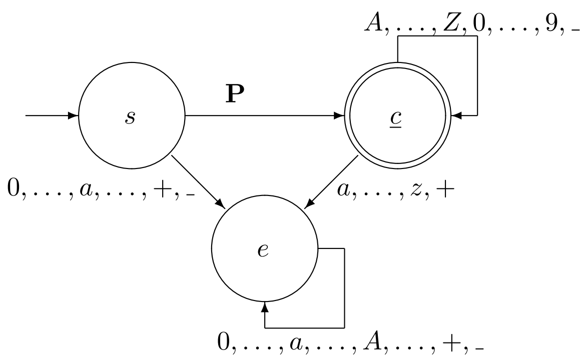


The word is read, we are in the final state, so the word $PI2$ is accepted.

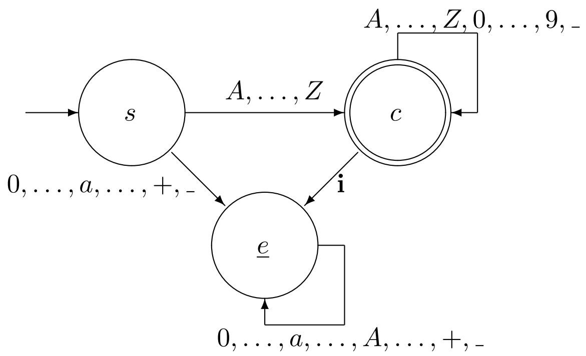
Let us now trace how the automaton will react to the word $Pi2$. We also start in the start state s:



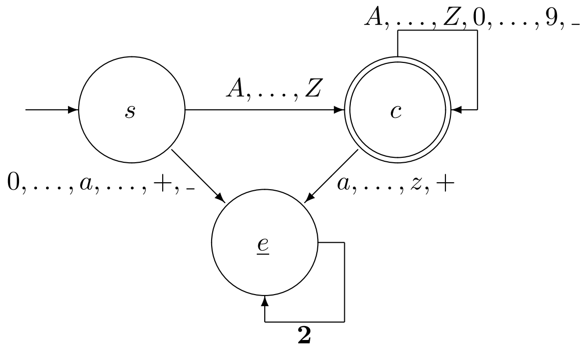
Then, we read the first letter P of the word $\mathbf{P}i2$, so we move to the state c;



After that, we read the second symbol i of the word $P\mathbf{i}2$ and move to state e:



Then, we read the last symbol 2 and stay in the state e:



We have read all the symbols, we are in the state e which is not final, so the word $Pi2$ is not accepted.

Task 1.3. Write down the tuple $\langle Q, \Sigma, \delta, q_0, F\rangle$ corresponding to the automaton from Part 1.1:

- Q is the set of all the states,
- Σ is the alphabet, i.e., the set of all the symbols that this automaton can encounter; for simplicity, consider only four symbols: the plus sign, letters a and A, and an underscore;
- $\delta : Q \times \Sigma \to Q$ is the function that describes, for each state q and for each symbol s, the state $\delta(q, s)$ to which the automaton that was originally in the state q moves when it sees the symbol s (you do not need to describe all possible transitions this way, just describe two of them);
- q_0 is the staring state, and
- F is the set of all final states.

Solution. $Q = \{s, c, e\}$, $\Sigma = \{a, A, +, \_\}$, $q_0 = s$, $F = \{c\}$, and the transition function δ is described by the following table:

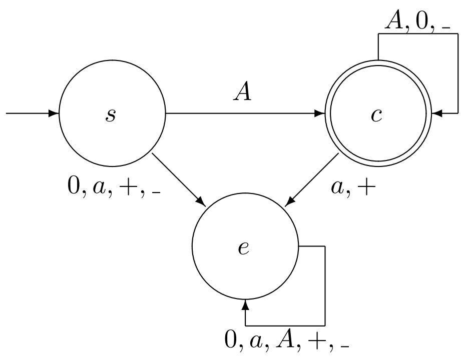
	$+$	a	A	$\_$
s	e	e	c	e
c	e	e	c	c
e	e	e	e	e

Task 1.4. Apply the general algorithm for union and intersection to:

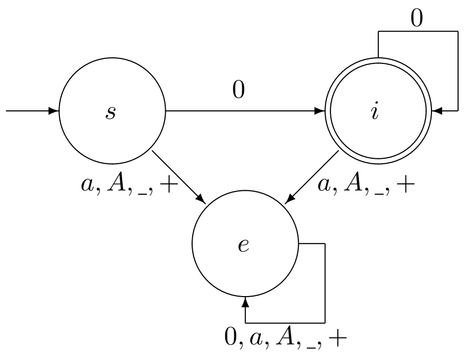
- the automaton from Part 1.1 as Automaton A and
- an automaton for recognizing unsigned binary integers – with which we started this class and which is described in the corresponding lecture – as Automaton B. In the example described in the lecture, we assumed, for simplicity, that, in addition to 0 and 1, only the symbol a is allowed; in reality, any other symbol different from 0 and 1 – including symbols A and underscore – leads to the error state e.

For simplicity, in your automaton for recognizing the union and intersection of the two languages, feel free to assume that you only have symbols 0, a, A, underscore, and +.

Solution. If we limit ourselves to these 5 symbols, then the Automaton A takes the following form:



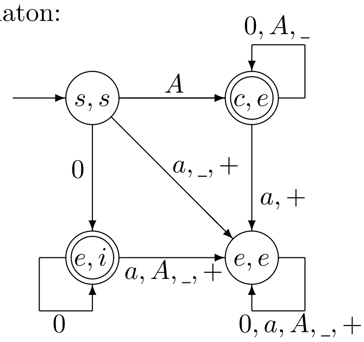
The Automaton B has the following form:



In the beginning, before we see any symbols, both automata are in the state s, so the combined automaton is in the state (s, s). Then:

- if we read A, Automaton A goes into state c and automaton B goes into state e, so we go into the state (c, e);
- if we read 0, then A goes into e and B goes into i, so the combined automaton goes into (e, i);
- if we read a, _, or $+$, then both automata go into e states, so the combined automaton goes into (e, e).

We can similarly describe transitions from these three new states. As a result, we get the following automaton:



Note that no state is final for both automata, so no state is final for the intersection.