14.

Evolutionary Computing - Module 2.

Darwian Evolution.

Given an environment that can host only a limited number of individual, and the basic instinct of individual to reproduce, selection become inevitable if the population size is not to grow exponentially.

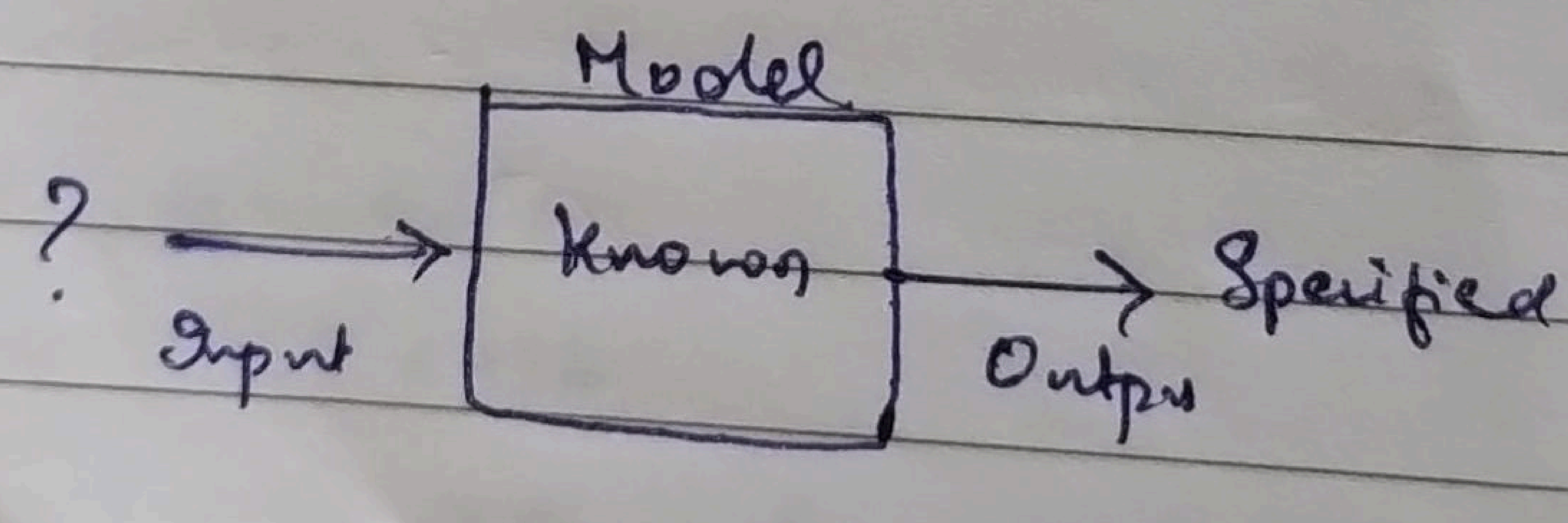
Why Evolutionary Computing.

-> Automated Problem Solver.

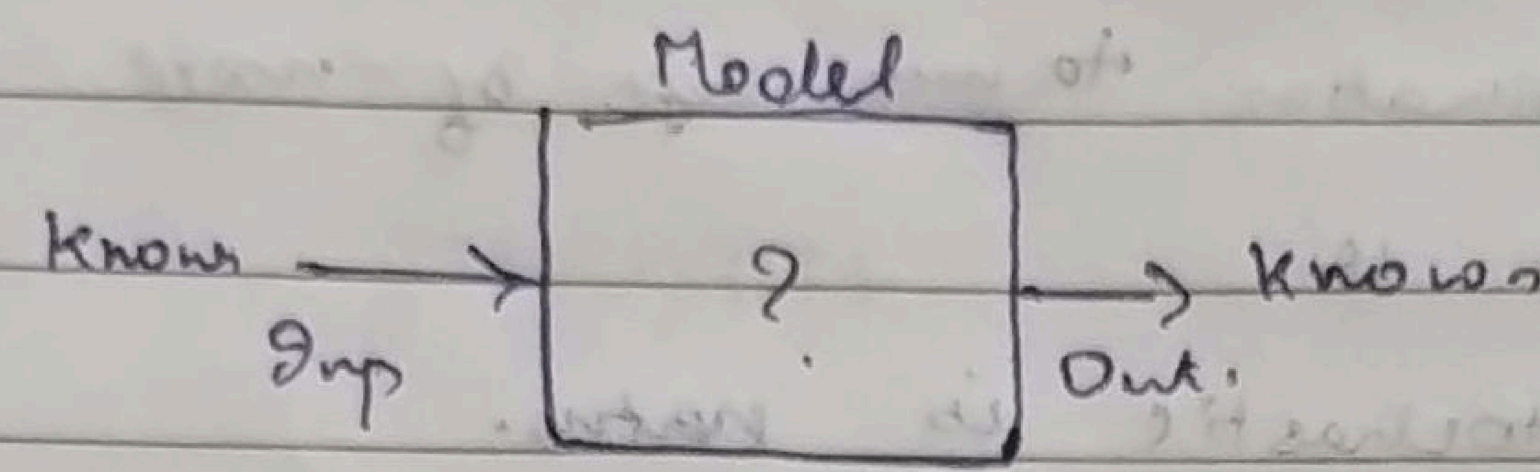
-> Copying Natural Problem Solver.

-> for ex: The human brain - neurocomputing.
Evolutionary Process - Evolutionary Computing

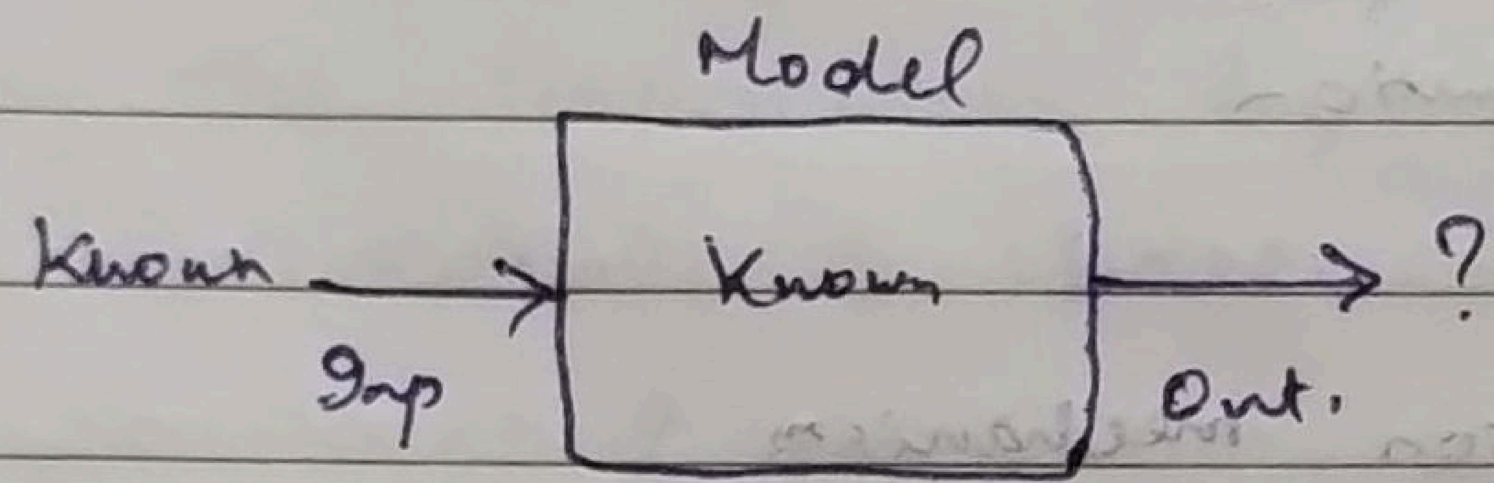
1. Optimization Problem
   Identify set of input.



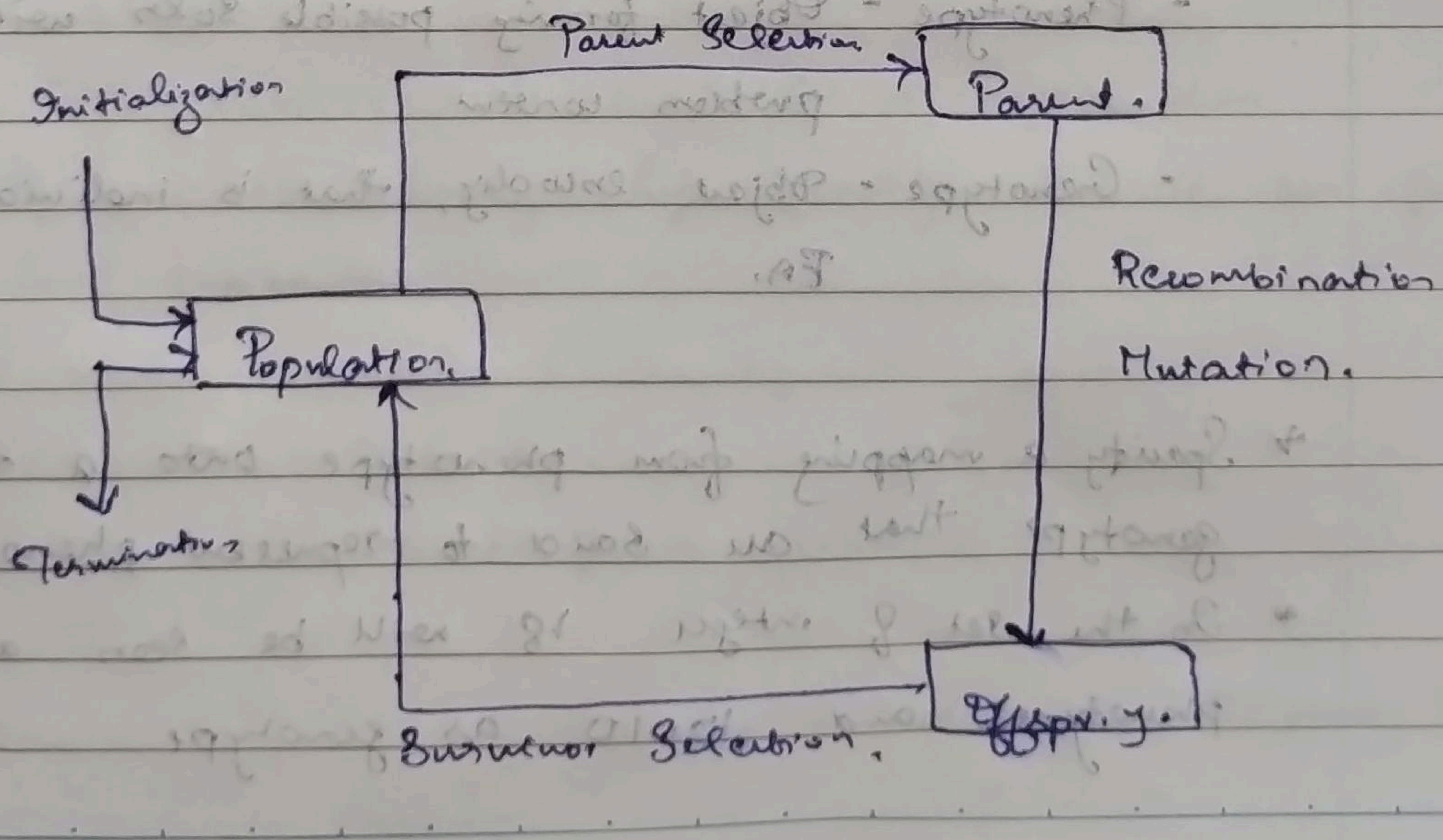
2 Modeling / System identification Problem



3. Simulation Problem.



-> There are different variations of evolutionary algorithm
The common idea is,
* Give a population of individual
* Environmental pressure causes natural Selection.



Properties.

* EA's are population based.
* Use recombination to mix info of more candidate solution into ce.
* They are stochastic in nature.

Components.

1. Representation
2. Evaluation function
3. Population
4. Parent selection mechanism
5. Variation operator, recombination and mutation
6. Survivor selection mechanism

1. Representation

* Link realworld to EA world.
  - Phenotype - Object forming possible solns within original problem context
  - Genotype - Object encoding, that is individual within EA.

* Specify a mapping from phenotype onto a set of genotype that are said to represent phenotype.
* In the set of integer 18 will be seen as phenotype and 10010 as genotype

2. Evolution function.

* It is a function / procedure that assign a quality measure to genotypes.

3. Population.
* To hold possible soln
* Multiset of genotype.
* Setting the population size.
* Best individual is chosen as seed for next gen
* The diversity of population is a measure of the no of different solns present.

4 Parent Selection Mechanism.
* To distinguish among parents.
* It is a parent if it has been selected to undergo variation
* high quality individual get a higher chance
* If we are choosing high quality then it can go into local optimum.

5. Variation operation
* New individual from old us
  → Mutation.
    ↳ One genotype and deliver a slightly modif mutant.

→ Recombination.

↳ A binary variation operator

Merges info from 2 parent genotype into 1 or 2

6.5 Survivor Selection

* Distinguish among individual based on their quality
* Survivor selection is also often seen.

I Initialization

* Randomly generated individual.
* Problem specific heuristic initial population with high fitness

II Termination

* has a known optimum fitness level coming from a known optimum of the given objective function then reaching this level is stopping condition.

* Max allowed CPU time elapse
* Total no of fitness evaluation reaches a given limit.
* for a given period of time - for a no of generation or fitness evaluation
* Diversity drop under a given threshold

Date :

22/8/24
Thursday

Genetic Algorithm:

* developed by John holland in 1960's and 1970's.
* representation -> bitstring.
* Mutation is bitflip.

Representation of Individual.

1. Binary Representation.

* Mutation for binary representation

1 0 1 0 0 0 0 1 0 → 1 0 0 1 0 0 0 0 0

2. Integer Representation.

* Random Resetting

-> Will have a probability P_m which is chosen at random from set of permissable value in each position

* Creep Mutation.

-> To make small change relative to range of possible value.
-> for ordinal attributes and work by adding a small value to each gene with probability P

3. Floating Point.
-> change gene randomly within its domain.

* Uniform Mutation
=> uniformly randomly from $[L_i, U_i]$.
=> Position wise mutation prob
-> Analogous to bit flipping for binary encoding and random resetting sketched for integer encoding.

29/8/24
Thursday

* Non uniform mutation with fixed distribution.
-> Analogous to the creep mutation.
-> Adding to the current gene value an amount drawn randomly from a gaussian distribution with mean zero and user specified standard deviation and then curtailing the resulting value to the range $[L_i, U_i]$ if necessary.

4. Mutation Operators for Permutation representation

1. Swap Mutation -> cannot use a random
2. Insert Mutation mutation operation
3. Scramble Mutation.
4. Inversion Mutation.

Recombination

-> new individual from information from 2 or more parent.

1. Binary Representation.

    1. One point crossover.

    2. N-point cross over.

    3. Uniform crossover.

        -> If value is below parameter p, the gene is taken from 1st parent or else from second parent

2. Integer Represent

    1. Same as binary representation.

3. Floating Point Representation:

2/9/24 Monday

    1. Arithmetic Recombination.

        -> Three types.

            a. Simple Recombination

    2. Single Arithmetic Recombination.

        -> Pick a gene or cell take arithmetic avg of 2 parent

    3. Whole Arithmetic Recombination.

        -> Take weighted sum of two parental allele for each gene.

Child 1 : $\alpha \cdot \bar{x} + (1-\alpha) \cdot \bar{y}$, child 2 = $\alpha \bar{y} + (1-\alpha) \bar{x}$

23

Choice	Element Selected	Reason	Partial Result.
All	1	Random.	[1]
2,5,4,9	5	Shortest list	[1, 5]
4, 6	6.	Common edge	[1 5 6]
2, 7	2	Random choice	[1 5 6 2]
3, 8	8	Shortest list	[1 5 6 2 8]
7, 9	7	Common edge	[1 5 6 2 8 7]
3	3	Only item in list	[1 5 6 2 8 7 3]
4, 9	9	Random choice	[1 5 6 2 8 7 3 9]
4	4	last element	[1 5 6 2 8 7 3 9 4]

C A B D E F

A B C E F D.

Element	Edge.
~~A~~	~~B F D, C~~
~~B~~	~~A F C, D~~
~~C~~	~~B, E, A, F~~
D	~~F~~, ~~A~~, ~~B~~, ~~E~~
~~E~~	~~D, F F, C~~
~~F~~	~~E F C, D~~

$\boxed{A}$

Choice	Element Selected	Reason	Partial Result
All	A	Random choice	[A]
B, D, C	B	Common edge	[A B]
C, D	C	Random choise	[A B C]
E, F	E	~~Random choice~~	[A B C E]
D, F	F	Common edge	[A B C E F]
D	D	last element	{A B C E F D}