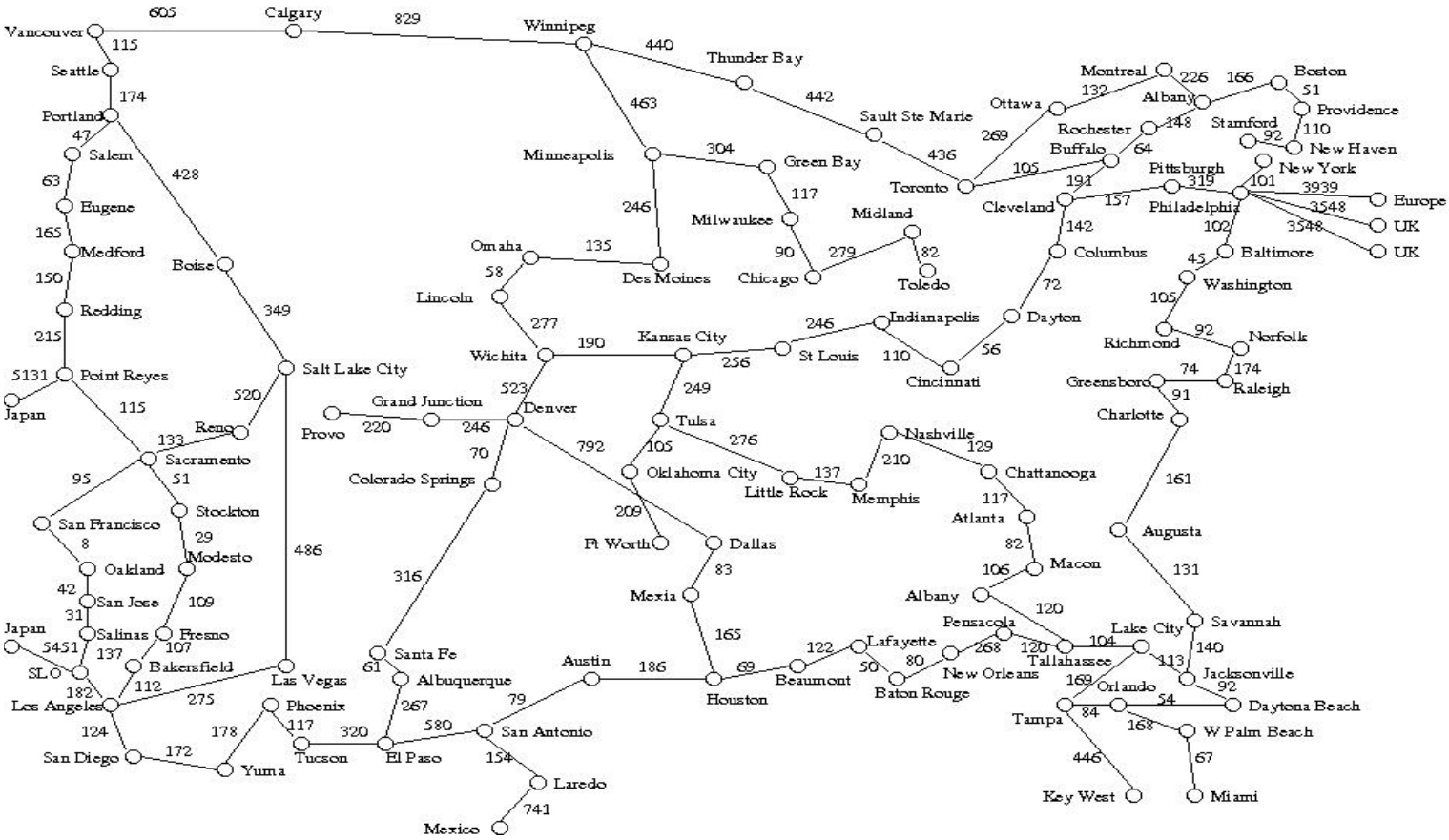
ASSIGNMENT #02 CSC 520-001/601 FALL 2017

[Due: before 2345 T 26 SEP]

Name: AKSHAY NALWAYA
Student ID: 200159155

1. This question concerns route-finding, with comparison of several search algorithms. This time, we're in the U.S. Here's jpg of the map below.
The solution consists of the series of cities a network packet must pass through, each city connected to one or more others by network links of the indicated length. There are no other network links.
In a language of your choice (Java or C++), implement the A* search algorithm. Your code should keep track of nodes expanded and should be able to compute the number of such nodes. Then run your algorithm on the telecom link map.



- Name the source code file with the main function `SearchUSA`, with the file extension appropriate to the language.
- The inputs should will be given through the command line, similar to Assignment 1.
    - In java:
      % java SearchUSA searchtype srccityname destcityname
    - In C++:
      % mv ./a.out SearchUSA
      % SearchUSA searchtype srccityname destcityname
- The searchtype should be either `astar`, `greedy`, or `uniform`.
- To save a bit of typing, the network is implemented as Prolog procedures in usroads.pl. This is a Prolog source file, and this assignment does not use Prolog. The file is provided solely as a convenience; you will have to modify it for use with your code. This time we will need the distances, and the longitude/latitude of the cities. (The percent sign (%) is a Prolog

comment char, from there to end-of-line.)

- The spelling of `srccityname` and `destcityname` must be the as given in usroads.pl. Do NOT change the names from lower case to upper case.
- A node is said to be **expanded** when it is taken off a data structure and its successors generated.
- Use as a heuristic the straight line distance between cities. The straight-line distance between cities is computed using decimal degrees of latitude and longitude, which also given in the file. There is a complication in computing straight line distance from longitude and latitude that arises because the earth is roughly a sphere, not a cylinder. As a guide, heuristic.pl is another Prolog file with a header comment indicating how the heuristic should work.
- As in Assignment 1, test your code on remote linux machines. Your code must compile without errors (warnings are okay) and must not print statements other than the ones stated above.

Now perform some experiments.

The experiments

1. **[10 points] Experiment with executing your implementation of A* to find various paths, until you understand the meaning of the output. Are there any pairs of cities (A,B) for which the algorithm finds a different path from B to A than from A to B? Are there any pairs of cities (A,B) for which the algorithm expands a different total number of nodes from B to A than from A to B?**

   Output in such cases should consist of the following:

   - **A comma separated list of expanded nodes (the closed list);**
   - **The number of nodes expanded;**
   - **A comma-separated list of nodes in the solution path;**
   - **The number of nodes in the path;**
   - **The total distance from A to B in the solution path.**

Ans.

A-star gives the same path for city A to city B and city B to city A but there is a difference in the number of nodes expanded during each case.

CASE 1 - astar from houston to omaha

```
hpc% java SearchUSA astar houston omaha
A-STAR SEARCH RESULT:
Expanded Cities: houston, mexia, dallas, beaumont, austin,
lafayette, batonRouge, sanAntonio, newOrleans, laredo,
pensacola, denver, coloradoSprings, tallahassee, elPaso,
albanyGA, wichita, albuquerque, macon, santaFe, atlanta,
chattanooga, nashville, lincoln, lakeCity,
Number of cities expanded: 25
Cities in final solution path: houston, mexia, dallas, denver,
wichita, lincoln, omaha,
Number of cities in solution path: 7
Total Distance(f) = 1898.0
```

CASE 2 - astar from omaha to houston

```
hpc% java SearchUSA astar omaha houston
A-STAR SEARCH RESULT:
Expanded Cities: omaha, lincoln, wichita, desMoines,
```

```
kansasCity, tulsa, oklahomaCity, ftWorth, littleRock,
minneapolis, stLouis, memphis, denver, coloradoSprings,
greenBay, milwaukee, chicago, dallas, mexia,
Number of cities expanded: 19
Cities in final solution path: omaha, lincoln, wichita,
denver, dallas, mexia, houston,
Number of cities in solution path: 7
Total Distance(f) = 1898.0
```

2. **[10 points] Change your code so as to implement greedy search, as discussed in the web notes.**

3. **[10 points] Do enough exploration to find at least one path that is longer using greedy search than that found using A*, or to satisfy yourself that there are no such paths. Find at least one path that is found by expanding more nodes than the comparable path using A*, or satisfy yourself that there are no such paths. If there is such a path, list the nodes in the path and the total distance.**

   Output in such cases should consist of the following:

   - **A comma separated list of expanded nodes (the closed list);**
   - **The number of nodes expanded;**
   - **A comma-separated list of nodes in the solution path;**
   - **The number of nodes in the path;**
   - **The total distance from A to B in the solution path.**

Ans.

The path from Yuma to Dayton using Greedy search has more number of expanded cities (35) and the number of cities in the final path (29), as compared to the same path for A-Star search method where 34 cities are present in expanded list while 14 cities in final path.

CASE 1: Greedy Search from yuma to dayton

OUTPUT:

```
hpc% java SearchUSA greedy yuma dayton
GREEDY BEST FIRST SEARCH RESULT:
Expanded cities: yuma, phoenix, tucson, elPaso, sanAntonio,
austin, houston, beaumont, lafayette, batonRouge, newOrleans,
pensacola, tallahassee, albanyGA, macon, atlanta, chattanooga,
nashville, memphis, littleRock, lakeCity, jacksonville,
savannah, augusta, charlotte, greensboro, raleigh, norfolk,
richmond, washington, baltimore, philadelphia, pittsburgh,
cleveland, columbus,
Number of cities expanded: 35
Cities in final solution path: yuma, phoenix, tucson, elPaso,
sanAntonio, austin, houston, beaumont, lafayette, batonRouge,
newOrleans, pensacola, tallahassee, lakeCity, jacksonville,
savannah, augusta, charlotte, greensboro, raleigh, norfolk,
richmond, washington, baltimore, philadelphia, pittsburgh,
cleveland, columbus, dayton,
Number of cities in solution path: 29
Total Distance(f) = 4191.0
```

CASE 2: A-Star Search from yuma to dayton

OUTPUT:

```
hpc% java SearchUSA astar yuma dayton
A-STAR SEARCH RESULT:
Expanded Cities: yuma, phoenix, tucson, elPaso, sanDiego,
albuquerque, santaFe, losAngeles, sanAntonio, austin,
lasVegas, bakersfield, coloradoSprings, houston, beaumont,
denver, lafayette, batonRouge, fresno, sanLuisObispo, mexia,
saltLakeCity, newOrleans, dallas, laredo, wichita, kansasCity,
modesto, stLouis, indianapolis, stockton, salinas, cincinnati,
sanJose,
Number of cities expanded: 34
Cities in final solution path: yuma, phoenix, tucson, elPaso,
albuquerque, santaFe, coloradoSprings, denver, wichita,
kansasCity, stLouis, indianapolis, cincinnati, dayton,
Number of cities in solution path: 14
Total Distance(f) = 2710.0
```

4. **[10 points] Change your code so as to implement uniform cost search, as discussed in the web notes.**

5. **[10 points] Do enough exploration to find at least one path that is longer using uniform cost than that found using A*, or to satisfy yourself that there are no such paths. Find at least one path that is found by expanding more nodes than the comparable path using A*, or satisfy yourself that there are no such paths. If there is such a path, list the nodes in the path and the total distance.**

   Output in such cases should consist of the following:

   - **A comma separated list of expanded nodes (the closed list);**
   - **The number of nodes expanded;**
   - **A comma-separated list of nodes in the solution path;**
   - **The number of nodes in the path;**
   - **The total distance from A to B in the solution path.**

Ans.

Uniform cost search always gives the same final path as that by A-Star search but the number of cities expanded are very high in case of uniform cost search as compared to A-star search.

CASE 1: A-Star search from cincinnati to winnipeg

OUTPUT:

```
hpc% java SearchUSA astar cincinnati winnipeg
A-STAR SEARCH RESULT:
Expanded Cities: cincinnati, indianapolis, dayton, columbus,
stLouis, cleveland, kansasCity, buffalo, pittsburgh, toronto,
rochester, saultSteMarie, wichita, lincoln, omaha, thunderBay,
tulsa, desMoines,
Number of cities expanded: 18
Cities in final solution path: cincinnati, dayton, columbus,
cleveland, buffalo, toronto, saultSteMarie, thunderBay,
```

```
winnipeg,
Number of cities in solution path: 9
Total Distance(f) = 1884.0
```

CASE 2: Uniform Cost search from cincinnati to winnipeg

OUTPUT:

```
hpc% java SearchUSA uniform cincinnati winnipeg
UNIFORM COST SEARCH RESULT:
Expanded Cities: cincinnati, dayton, indianapolis, columbus,
cleveland, stLouis, pittsburgh, buffalo, rochester, toronto,
kansasCity, albanyNY, philadelphia, wichita, ottawa, boston,
newYork, baltimore, tulsa, providence, washington, montreal,
oklahomaCity, richmond, newHaven, saultSteMarie, lincoln,
norfolk, stamford, omaha, littleRock, ftWorth, raleigh,
desMoines, memphis, denver, greensboro, coloradoSprings,
charlotte, thunderBay, nashville, minneapolis, grandJunction,
augusta, chattanooga, santaFe, savannah, atlanta, albuquerque,
provo, macon, greenBay, jacksonville,
Number of cities expanded: 53
Cities in final solution path: cincinnati, dayton, columbus,
cleveland, buffalo, toronto, saultSteMarie, thunderBay,
winnipeg,
Number of cities in solution path: 9
Total Distance(f) = 1884.0
```

6. **As part of your answer, compare the solution paths and explain what happened, especially any weird behavior you might detect**

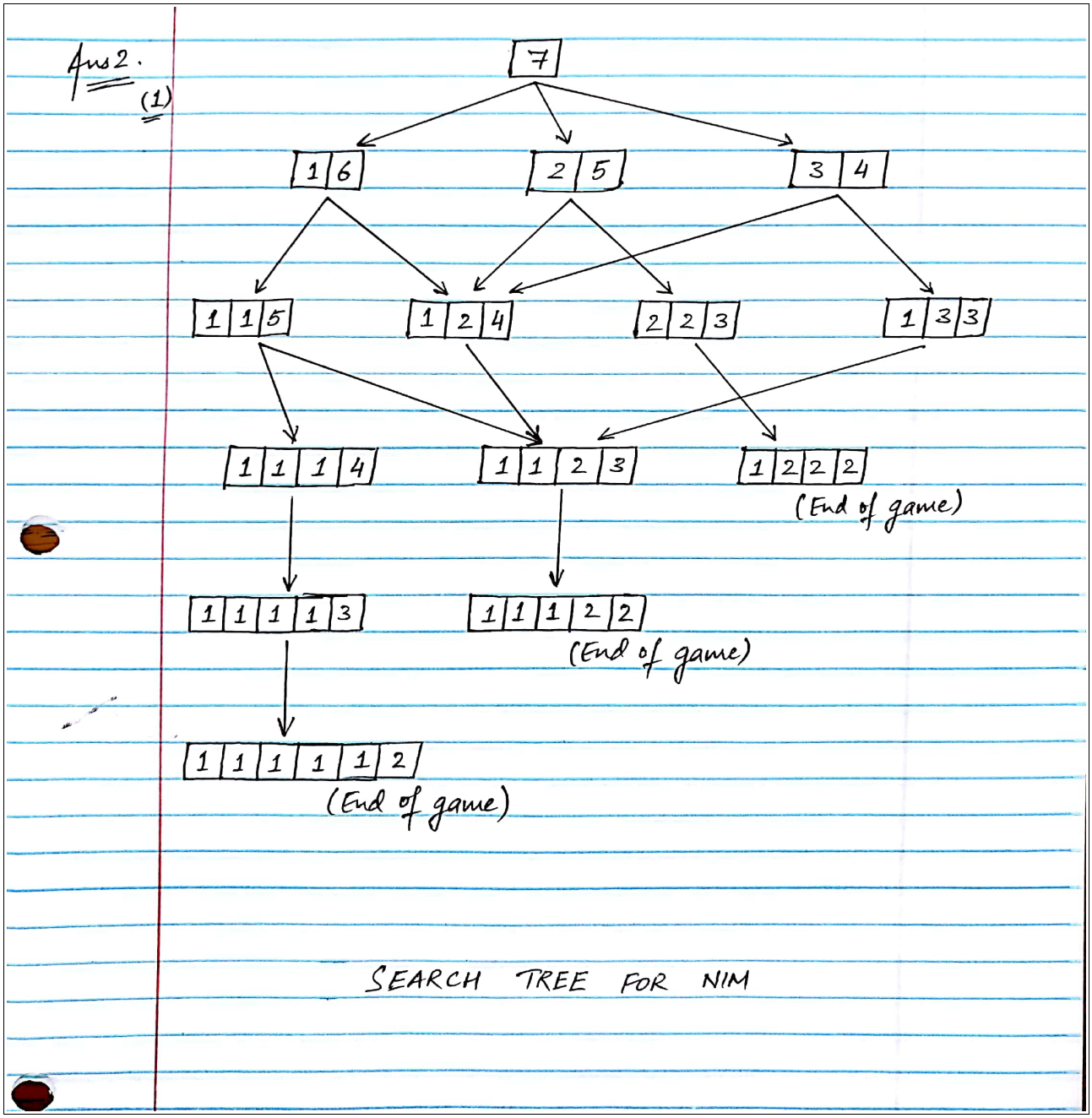
Ans.

- A-Star and Uniform Cost Search always gives the same path and hence the total path distance but in case of greedy best first search, the path length is different and is generally much higher than A-star or uniform cost search.
- Number of nodes expanded are much higher in case of uniform cost search as compared to A-star and greedy search, owing to the fact that uniform cost always explores in every direction and has no information about the goal location.
- Greedy best first search is not guaranteed to give the optimal path since it expands the node that seems to be the closest and does not take into consideration the value of 'f' (which is equal to g+h).

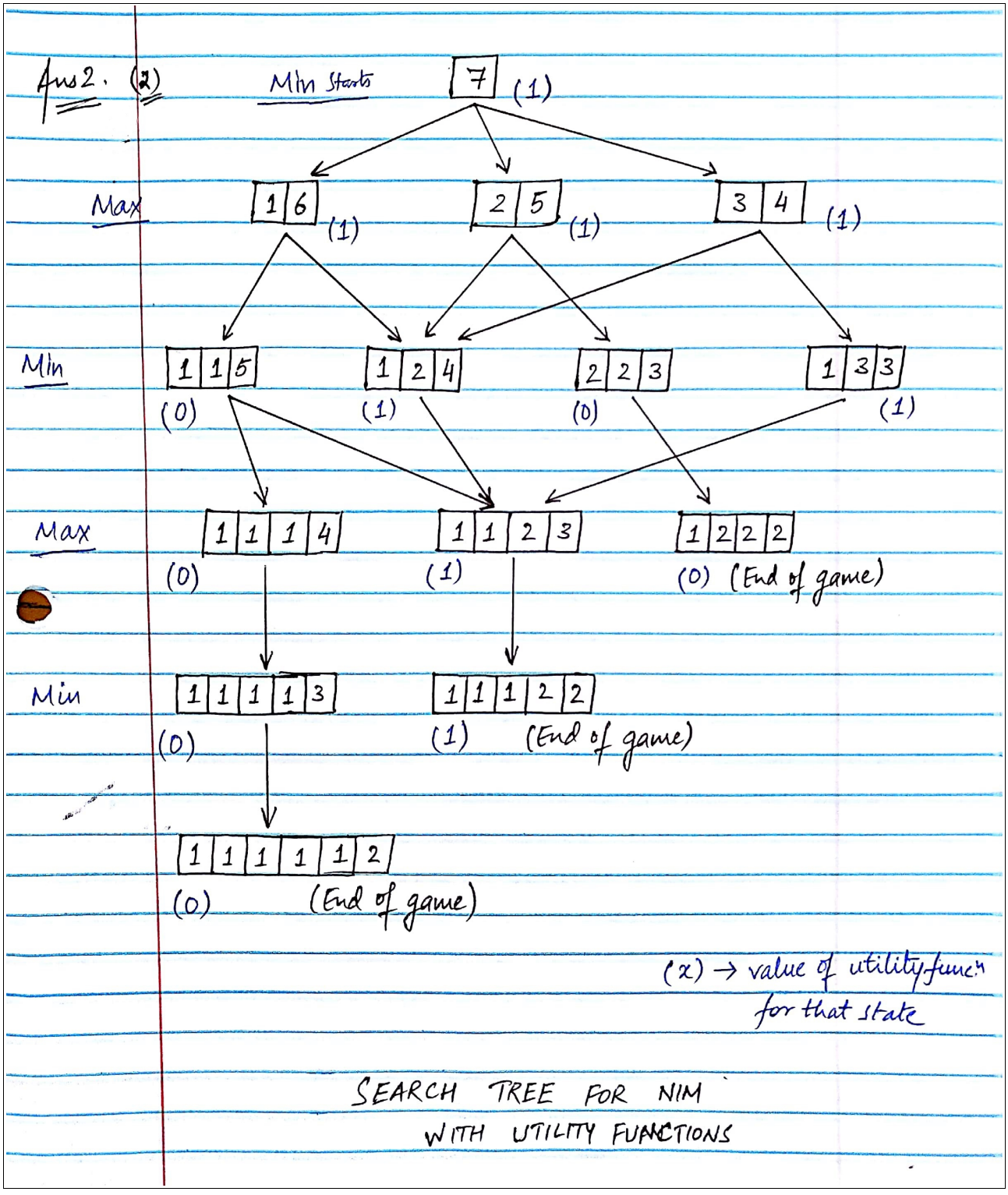
2. [15 points] Nim is a two-player game and the rules are as follows: The game starts with a single stack of 7 tokens. At each move a player selects one stack and divides it into two non-empty, non-equal stacks. A player who is unable to move loses the game.

1. **Draw the complete search tree for Nim.**
2. **Assume two players, Min and Max, play Nim and Min performs the first move. If a terminal state in the search tree developed above is a win for Min, a utility function of zero is assigned to that state. A utility function of 1 is assigned to a state if Max wins the game. Apply the min-max algorithm to the search tree to assign utility functions to all states in the search tree.**
3. **If both Min and Max play a perfect game (optimally), who will win? Explain your answer.**

Ans 2. (1)



Ans 2. (2)



Ans 2. (3)

If both Min and Max will play the game optimally, then for every case, ***Max will win*** as the utility function assigned to the root node (initial state) in search tree for Nim has the utility function value = 1, i.e., Min will never be able to take the path which terminates at a state where it will win.

3. [10 points] Convert the following set of sentences to Conjunctive Normal Form and use resolution to prove the conclusion: ¬R∧¬S

Try to find a shorter proof than the 4-Part Heuristic will give you.

1. **¬T∨¬Y⇒¬S**
2. **R⇔(S∨X)**
3. **X⇒U**
4. **T∧Y⇒¬S**
5. **X⇒S**

Ans 3.

Ans 3.

PL :

(1) $\neg T \vee \neg Y \Rightarrow \neg S$

(2) $R \Leftrightarrow (S \vee X)$

(3) $X \Rightarrow U$

(4) $T \wedge Y \Rightarrow \neg S$

(5) $X \Rightarrow S$

CNF : Conclusion : $\neg R \wedge \neg S$

(1) $T \vee \neg S$

(2) $Y \vee \neg S$

(3) $\neg R \vee S \vee X$

(4) $\neg S \vee R$

(5) $\neg X \vee U$

(6) $\neg X \vee U$

(7) $\neg T \vee \neg Y \vee \neg S$

(8) $\neg X \vee S$

(9) $R \vee S$ Negated conclusion

Applying 4-part heuristic to resolve them,

(10) $S \vee X$ [(9) + (3)]

(11) $X \vee T$ [(10) + (1)]

(12) $T \vee R$ [(11) + (5)]

(13) $R \vee \neg Y \vee \neg S$ [(12) + (7)]

(14) $\neg Y \vee X$ [(13) + (3)]

(15) $X \vee \neg S$ [(14) + (2)]

(16) $\neg S \vee R$ [(15) + (5)]

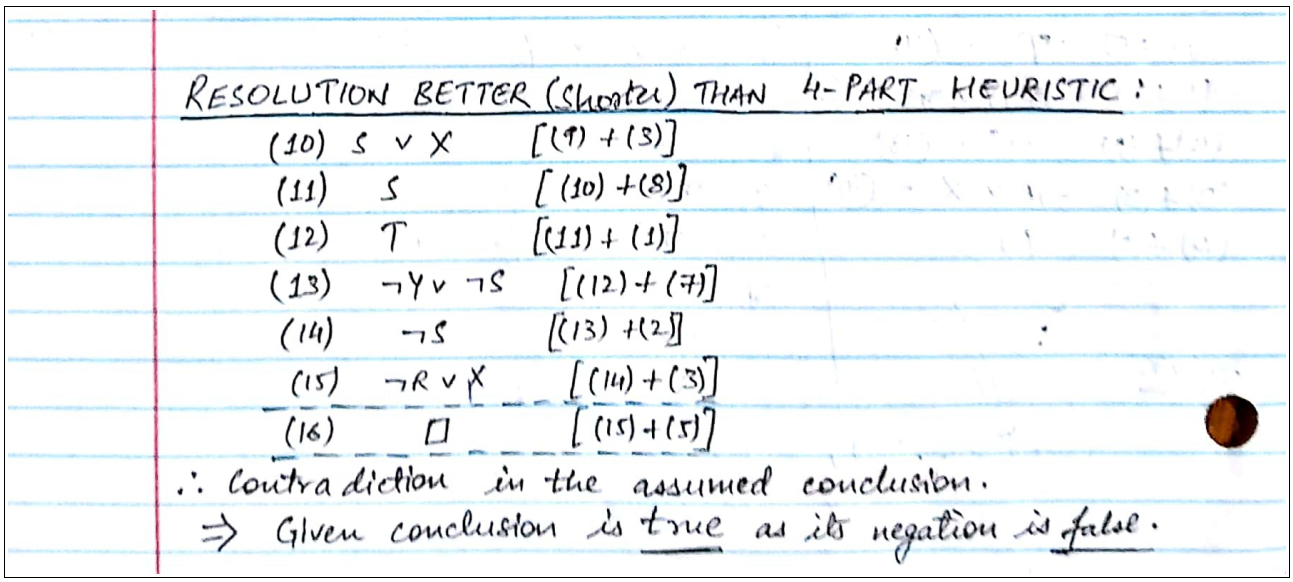
(17) X [(16) + (3)]

(18) R [(17) + (5)]

(19) $S \vee X$ [(18) + (3)] Same as (10)

∴ We will have to backtrack and choose some other clause after (3) to move out of infinite loop.

It is evident that 4-part heuristic will give a solution that generates clauses at least till clause #19, but there is another solution which gives us an *Empty Clause* in lesser number of steps.



Therefore, we can conclude that this approach is better than the solution which 4-part heuristic will give.

4. [10 points] Use Propositional Logic to determine whether or not the following set of requirements is logically consistent. In other words, represent the following sentences in Propositional Logic, convert to Conjunctive Normal Form, and run Resolution until a contradiction is derived, or else show *one* model of all the expressions showing that no contradiction exists.

The cat will play if it wants attention.
If it is in the evening, the cat wants dinner.
If the cat is meowing, it wants either dinner or attention.
If it is not in the evening, then the cat is meowing.
The cat does not want dinner.
The cat will not play.

Use the following lexicon:

- **m — The cat is meowing.**
- **a — The cat wants attention.**
- **w — The cat wants dinner.**
- **e — It is in the evening.**
- **p — The cat will play.**

Ans 4.

Ans 4.

Representing given sentences in PL,

$a \Rightarrow p$
$e \Rightarrow w$
$m \Rightarrow (a \vee w)$
$\neg e \Rightarrow m$
$\neg w$
$\neg p$

Converting these into Conjunctive Normal Form:

(1) $\neg a \vee p$
(2) $\neg e \vee w$
(3) $\neg m \vee a \vee w$
(4) $e \vee m$
(5) $\neg w$
(6) $\neg p$

Now, applying resolution,

(7) $\neg a$ [(6) + (1)]
(8) $\neg m \vee w$ [(7) + (3)]
(9) $w \vee e$ [(8) + (4)]
(10) e [(9) + (5)]
(11) w [(10) + (2)]
(12) $\square$ [(11) + (5)]

↘ "Empty set"

∴ Contradiction

⇒ The set of sentences are logically inconsistent.

5. [15 points] Write the following English sentences in First Order Logic, convert those First Order Logic sentences into CNF, and prove the conclusion by resolution and the 4-part heuristic presented in class. Number your clauses, and indicate explicitly step-by-step what resolves together, under what substitution.

Anyone whom Mary admires is a baseball star.
Any freshman who does not pass the exam does not play.
John is a freshman.
Any freshman who does not study does not pass the exam.
Anyone who does not play is not a baseball star.
(Conclusion) If John does not study, then Mary does not admire John.

Use the following lexicon:

Admire(X,Y) -- X admires Y
BaseballStar(X) -- X is a baseball star
Freshman(X) -- X is a Freshman
Play(X) -- X plays
Study(X) -- X studies
Pass(X) -- X passes the exam

Ans 5.

<<solution on next page>>

Ans 5.

Ans 5.

FOPL :

① $\forall X\,(Admire(Mary, X) \Rightarrow BaseballStar(X))$

② $\forall X((Freshman(X) \wedge \neg Pass(X)) \Rightarrow \neg Play(X))$

③ $Freshman(John)$

④ $\forall X\,((Freshman(X) \wedge \neg Study(X)) \Rightarrow \neg Pass(X))$

⑤ $\forall X\,(\neg Play(X) \Rightarrow \neg BaseballStar(X))$

Conclusion :

⑥ $\neg Study(John) \Rightarrow \neg Admire(Mary, John)$

CNF :

(1) $\neg Admire(Mary, X1) \vee BaseballStar(X1)$

(2) $\neg Freshman(X2) \vee Pass(X2) \vee \neg Play(X2)$

(3) $Freshman(John)$

(4) $\neg Freshman(X3) \vee Study(X3) \vee \neg Pass(X3)$

(5) $Play(X4) \vee \neg BaseballStar(X4)$

---

(6) $\neg Study(John)$ Negated Conclusion

(7) $Admire(Mary, John)$ Negated Conclusion

(8) $BaseballStar(John)$ [(7) + (1) under {John/X1}]

(9) $Play(John)$ [(8) + (5) under {John/X4}]

(10) $\neg Freshman(John) \vee Pass(John)$ [(9) + (2) under {John/X2}]

(11) $Pass(John)$ [(10) + (3) under {}]

(12) $\neg Freshman(John) \vee Study(John)$ [(11) + (4) under {John/X3}]

(13) $Study(John)$ [(12) + (3) under {}]

(14) $\square$ [(13) + (6) under {}]

Since we get contradiction, so, conclusion is TRUE.