Answers of Mid-semester Examination:

Section 1:

2.

- State: represented as a tuple (x, y) where 0<=x<=5 and 0<=y<=4
- Initial State: (0, 0)
- Successor function: The successor state is generated by the following actions:
  Fill any jug
  Empty any jug
  Pour water from jug A to jug B until either jug A is empty or jug B is full
  Pour water from jug B to jug A until either jug B is empty or jug A is full
- Goal state: (0, 2) or (2, 0)

3. a.

K=0
OL={S}
OL= {}

K=1

OL={S}
OL= {A, B, D}
OL= {B, D}
OL = {D}
OL = {}

K=3
OL={S}
OL= {A, B, D}
OL= {E, G, B, D}
OL = {G, B, D}
OL = {B, D}

Goal Node found!!

b. Number of nodes generated=1+4+6

c. Solution path: S->A->G

Path cost: 22

Note: [4+1+1]; No marks for b and c without demonstration; there may be other paths also.

Section 2:

1.

I. OL={S(21)}

CL= {}

II. OL={A(11),B(13),C(25)}

CL={S (21)}

III. OL={B(13),D(21),C(25)}

CL={S(21), A(11)}

IV. OL={D(20),C(25)}

CL={S(21),A(11),B(13)}

V. OL={C(25),F(36),K(30), E(33)}

CL={S(21),A(11),D(20),B(13)}

VI. OL={D(17),F(36),K(30),E(33) }

CL={S(21),A(11),C(25),B(13)}

VII. OL={K(27),E(30),F(33)}

CL={S(21),A(11),B(13),C(25),D(17)}

VIII. OL={E(30),F(33),G(33)}

CL={S(21),A(11),B(13),C(25),D(17),K(27)}

IX. OL={G(31),F(33)}

CL={S(21),A(11),B(13),C(25),D(17),K(27),E(30)}

X. OL={F(33)}

CL={S(21),A(11),B(13),C(25),D(17),K(27),E(30),G(31)}

XI. OL={}

CL={S(21),A(11),B(13),C(25),D(17),K(27),E(30),G(31),F(33)}

Number of nodes expanded=10

Solution path:- S->C->D->E->G

Path cost = 31

b.
0<h(C)<=12
0<h(S)<=min(11,h(C)+5)

Note: [4+1+1] + [2]; No marks without demonstration; Node reopening is required so we need to expand till OL is empty.

Section 3:
Justify the following statement for AI search problem:

a. Node reopening is not needed for uniform cost search for the state space with positive arc cost.

b. A* search technique is optimal for consistent heuristic.

c. Greedy best first search is not optimal and complete for infinite state space.

d. Problem relaxation is helpful to design admissible heuristic.

e. DFS is space efficient than BFS.

f. Bidirectional search is not applicable for all type of AI problem.

g. Depth limited search may not be complete.

Answer:

a. UCS with positive arc cost is optimal (first goal node selected for expansion must be the optimal solution). In UCS with positive arc cost, all nodes are expanded once (in a optimal path; no node reopening).

b. A* search is optimal with consistent heuristic. In this case, node reopening is not needed (always expand node with optimal path). Here, f-value never decreases along the path and A heuristic h is consistent (or monotone) if h(N) <=w(N,N') + h(N') (for each node N and each successor N' of N).

c. For infinite state space, Greedy best first search is not optimal because it is heuristic driven and cannot guarantee to return the minimum-cost path for goal node.

   For infinite state space, Greedy best first search is not complete because it may enter in a wrong path and cannot able to come back to the correct path (leading to goal node) since the state space is infinite.

d. Standard approach to create a heuristic is problem relaxation. It will add some new actions for the problem so that search is not required to find the solution cost in the relaxed problem. Cost of optimal solution to the relaxed problem is an admissible heuristic for the original problem. It never overestimates the cost for reaching goal node from any node.

e. DFS needs to store a single path along with the remaining unexpanded sibling nodes for each node on the path. DFS requires a storage for only O(bm) nodes; whereas in BFS, we need to store all the nodes after expansion at each and every depth, resulting in the space complexity as $O(b^d)$. DFS is space efficient than BFS.

f. Bidirectional search is not applicable for all type of AI problem under the following cases:

   i. The complete description of goal state is not available always.
   ii. The designing of reverse action is not always possible.

g. Let, the depth cutoff be k and the goal nodes are present at depth d (deeper than k). In this case, Depth limited search will always incomplete for d>k, since it will not expand any nodes after depth k.