Information for Midterm 1

List ADT operations (pa1):

```
List newList();
void freeList(List* pL);
int length(List L);
int position(List L);
ListElement front(List L);
ListElement back(List L);
ListElement get(List L);
bool equals(List A, List B);
void clear(List L);
void set(List L, ListElement x);
void moveFront(List L);
void moveBack(List L);
void movePrev(List L);
void moveNext(List L);
void prepend(List L, ListElement data);
void append(List L, ListElement data);
void insertBefore(List L, ListElement data);
void insertAfter(List L, ListElement data);
void deleteFront(List L);
void deleteBack(List L);
void delete(List L);
void printList(FILE* out, List L);
List copyList(List L);
List join(List A, List B);
List split(List L);
```

Graph ADT operations (pa2):

```
Graph newGraph(int n);
void freeGraph(Graph* pG);
int getOrder(Graph G);
int getNumEdges(Graph G);
int getNumArcs(Graph G);
int getSource(Graph G);
int getParent(Graph G, int u);
int getDist(Graph G, int u);
void getPath(List L, Graph G, int u);
void makeNull(Graph G);
void addEdge(Graph G, int u, int v);
void addArc(Graph G, int u, int v);
void BFS(Graph G, int s);
void printGraph(FILE* out, Graph G);
```

Graph ADT operations (pa3):

```
Graph newGraph(int n);
void freeGraph(Graph* pG);
int getOrder(Graph G);
int getNumEdges(Graph G);
int getNumArcs(Graph G);
int getParent(Graph G, int u);
int getDiscover(Graph G, int u);
int getFinish(Graph G, int u);
void makeNull(Graph G);
void addEdge(Graph G, int u, int v);
void addArc(Graph G, int u, int v);
void DFS(Graph G, List S);
Graph copyGraph(Graph G);
Graph transpose(Graph G);
void printGraph(FILE* out, Graph G);
```

Graph Algorithms:

```
BFS(G,s)
1.  for x in V(G)-{s}
2.     color[x] = white
3.     d[x] = inf
4.     p[x] = nil
5.  color[s] = gray
6.  d[s] = 0
7.  p[s] = nil
8.  Q = { }
9.  Enqueue(Q,s)
10. while Q ≠ { }
11.    x = Dequeue(Q)
12.    for y in adj[x]
13.       if color[y] == white
14.          color[y] = gray
15.          d[y] = d[x]+1
16.          p[y] = x
17.          Enqueue(Q,y)
18.    color[x] = black
```

```
PrintPath(G,s,x)
1. if x == s
2.    print(s)
3. else if p[x] == nil
4.    print(x," is not reachable from ",s)
5. else
6.    PrintPath(G, s, p[x])
7.    print(x)
```

```
DFS(G)
1. for all x in V(G)
2.    color[x] = white
3.    p[x] = nil
4. time = 0
5. for all x in V(G)
6.    if color[x] == white
7.       Visit(x)
```

```
Visit(x)
1. d[x] = (++time)
2. color[x] = gray
3. for all y in adj[x]
4.    if color[y] == white
5.       p[y] = x
6.       Visit(y)
7. color[x] = black
8. f[x] = (++time)
```