Mini-project report

Group 8 – ICT

# Member

Nguyễn Thành Long 20180128

Chu Mạnh Hải 20184253

Nguyễn Mạnh Tiến 20184312

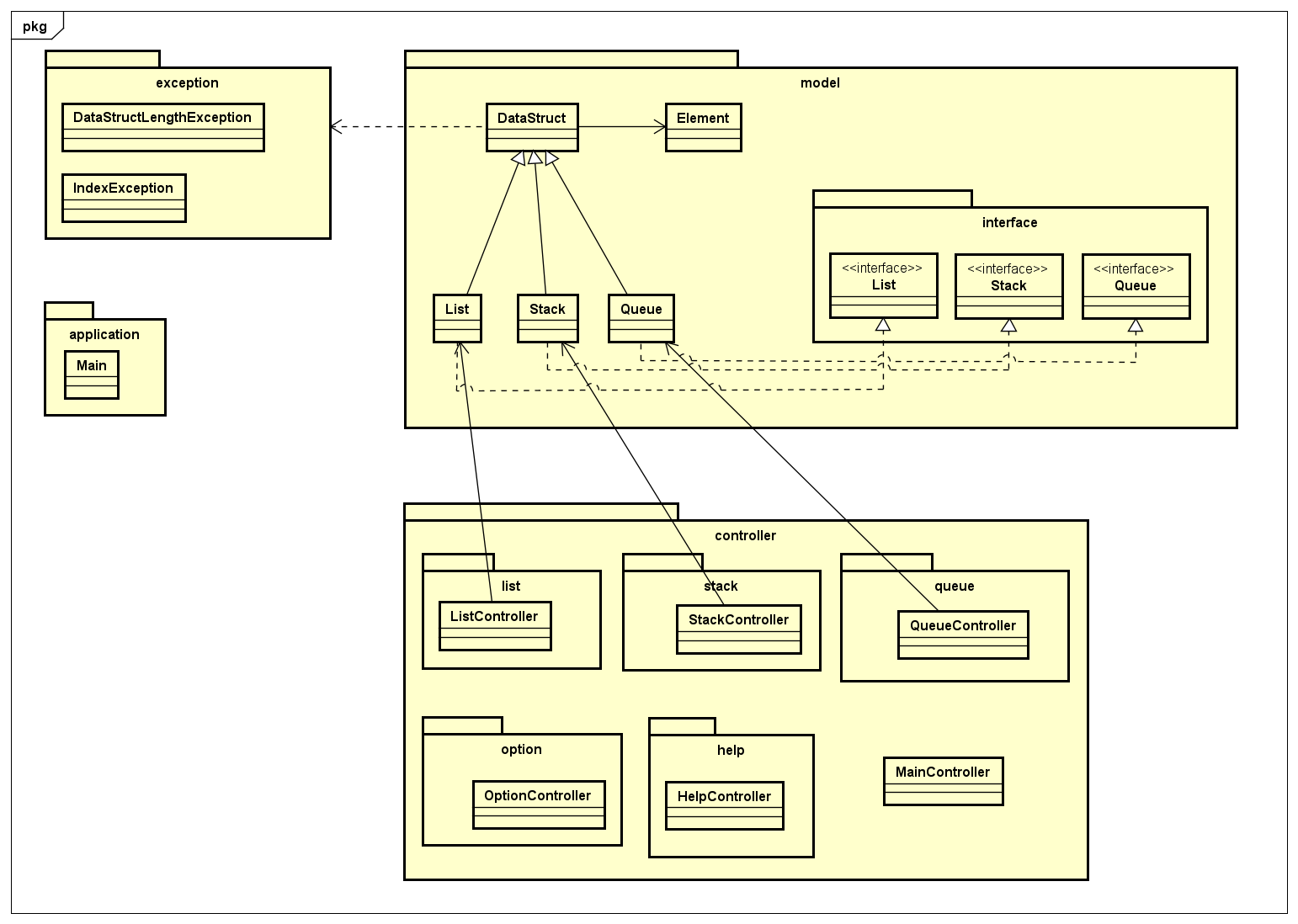
# Assignment of members

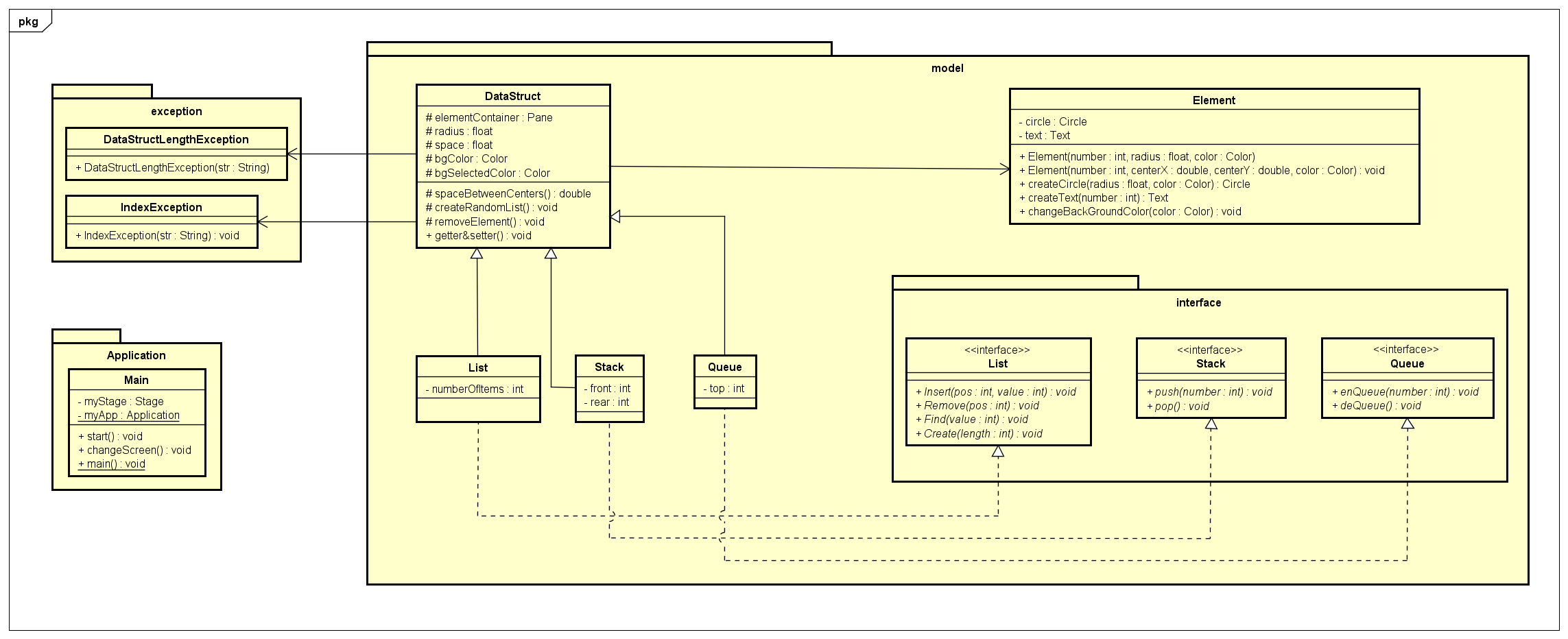
* DataStruct
  + protected void createRandomList(int length)
  + protected void insertElement(boolean isSeq, int value, int pos)
  + protected void removeElement(boolean isSeq, int pos)
  + protected double spaceBetweenCenters()
  + protected SequentialTransition moveElementVertical(boolean fadeOut, double toTranY, Element element)
  + protected ParallelTransition moveTheRestHorizontal(double startX, int idx)
  + public void resetData()
* Element
  + private Circle createCircle(float radius, Color color)
  + private Text createText(int number)
  + public void changeBackGroundColor(Color color)
  + public Color getBgColor()
  + public int getValue()
* List
  + public void Insert(int pos, int value)
  + public void Remove(int pos)
  + public int Find(int value)
  + public void Create(int length)
  + public boolean IsEmpty()
* Queue
  + public void enQueue(int number)
  + public void deQueue()
  + public void createRandomList(int length)
  + public int getRear()
  + public int getFont()
* Stack
  + public void push(int number)
  + public void pop()
  + public void createRandomList(int length)
  + public int getTop()

# Mini-project description

* Requirement: Simulate how linked list, stack and queue works:
  + Linked list:
    - Create linked list
    - Find the first element in list with input value
    - Insert an element into any position in the list
    - Remove element in any position in the list
  + Stack
    - Create stack
    - Push an element into stack
    - Pop an element from stack
  + Queue
    - Create queue
    - Enqueue an element
    - Dequeue an element
* Use case diagram:
  + Get help: See the instruction how to use the app
  + Choose type of data structure simulator: User choose 3 type of data structures are linked list, stack and queue. Then program will switch to corresponding screen
  + Create list: User enters the length then program creates a list with random element from input length
  + Insert element into list: User enters the value of new element and the position to insert it into, then program visualizes how linked list insert the new element
  + Find element in list: User enters a value to find in list, then program visualizes the algorithm to find the element and notice whether element’s found or not
  + Remove element from list: User enters the index of element to remove, then program visualizes how linked list find and remove that element
  + Create stack: User enters the length then program creates a stack with random element from input length
  + Push element into stack: User enters the value of new element, then program visualizes how push method works and how stack is change
  + Pop element from stack: The program visualizes how pop method works and how stack is change
  + Create queue: User enters the length then program creates a queue with random element from input length
  + Push element into queue: User enters the value of new element, then program visualizes how enqueue method works and how queue is change
  + Pop element from queue: The program visualizes how dequeue method works and how queue is change

# Design





* Relationships between classes:
  + List, Queue, Stack extend DataStruct
  + List implement IList
  + Queue implement IQueue
  + Stack implement IStack
* Implementation of important methods:
  + createRandomList(): create elements object and append to list to display
  + insertElement(): go through element in list, change color, until meet the position, stop and insert new element
  + removeElement(): go through element in list, change color, until meet the position, stop and remove element