**Temasek Polytechnic**

**School of Informatics & IT**

**Data Structures and Algorithms (CGE2C10)**

**AY 13/14 Apr Semester**

*Started* **Week 3** *Due* **Week 11 in lab**

# Assignment – Part 1 (30%)

### Introduction / Aim

This assignment requires students to apply the knowledge on data structures, such as stacks, queues, and linked lists. This is an **individual** assignment.

1. The undo button of Microsoft Word 2013 works by keeping track of all actions performed and reproducing the actions in a first-in-last-out fashion when clicked. Write an application called UndoButtonApp that will emulate a simple undo button functionality of Microsoft Word 2013.

Your program should comprise an Action class that contains only one attribute and its corresponding properties: actionName(String). The Action class has a constructor that takes in the actionName attribute.

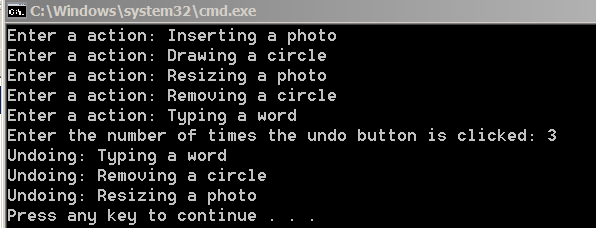
The main program will do the following:

1. Prompt the user to enter 5 action names. Create an Action object for each action name entered and insert the Action object onto a stack.
2. Prompt the user to enter the number of times the Undo button is clicked.
3. Emulate the undo button by displaying the corresponding action name for each click.

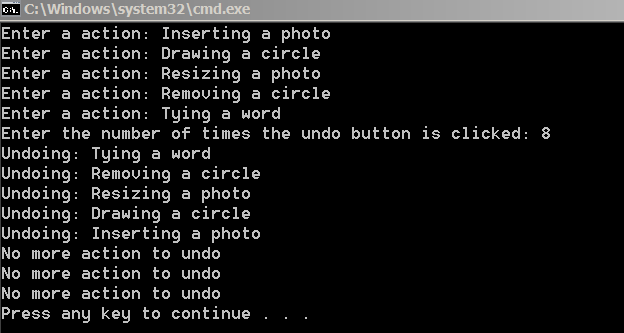
* When there are no more Action object left in the stack, display the text “No more action to undo”. This scenario will happen if the number of action performed is less than the number of times the undo button is clicked.
* You can refer to the samples below for the expected outputs.

**Samples of Expected Outputs:**

*When the number of undo button clicks is less than or equal to the number of actions performed:*



*When the number of undo button clicks is more than the number of actions performed:*



1. At Tampines Polytechnic, students are allocated their choice of cross-disciplinary subject (CDS) on a first-come-first-served basis. There are 3 CDSes offered: Music, Creative Writing and Theatrical Studies. Each CDS has a class size of 3. If the student is not able to get his/her selected CDS on his/her turn, he/she will need to rejoin the queue for another selection. You have been assigned to simulate this process by creating an application called CDSSellectionApp. Your program will do the following:
2. Prompt the user to enter the names of 9 students and add the names to a queue on a first-come-first-served basis. This will form the queue for the CDS selection process.
3. Once the CDS selection is in process, dequeue a student and prompt him/her for his/her CDS selection.
4. On a successful match, your program will inform the student that his/her application is successful.
5. However, if his/her selected CDS has no more vacancy, he/she will need to rejoin the queue.
6. Your program will continue to allocate CDS to students (steps b to d) until all students are successful in their CDS allocation.

//3 queues

1 student queue.

2nd round is the left over

**Sample Output:**

|  |
| --- |
| Welcome to the CDS Selection Application. You can join the queue by entering your name. On your turn, please select 1 for Music, 2 for Creative Writing and 3 for Theatrical Studies. Have a pleasant day!  Enter the name of student #1: Kai Fong  Enter the name of student #2: Angel  Enter the name of student #3: Damian  Enter the name of student #4: Wei Jun  Enter the name of student #5: Debbie  Enter the name of student #6: Ting Xing  Enter the name of student #7: Jennifer  Enter the name of student #8: Peggy  Enter the name of student #9: Frances  #1 Round – CDS Selection in Process ….  Hi Kai Fong,  Please enter your choice of CDS: 2  Your application is successful!  Hi Angel,  Please enter your choice of CDS: 2  Your application is successful!  Hi Damian,  Please enter your choice of CDS: 2  Your application is successful!  Hi Wei Jun,  Please enter your choice of CDS: 2  Your application is **unsuccessful**! You will rejoin the queue for another selection.  Hi Debbie,  Please enter your choice of CDS: 1  Your application is successful!  ……  #2 Round – CDS Selection in Process ….  ……  End of CDS Selection Process…. |

3. Adam owns the EveryThingCanPack storage company where customers can send their parcels for a fee. Adam has decided to improve his company’s productivity by purchasing a software that could track the movement of his customers’ parcels. As the software developer in charge of delivering this software, you have been tasked to fulfil the following requirements:

1. Create a project called EveryThingCanPackAppand add the LinkedList.cs and Node.cs files to the project.
2. Create a Parcel class that contains three attributes and their properties: id(int), customerName(String) and weight(int). The Parcel class has a constructor that takes in all attributes.
3. Modify the LinkedList class to include the following 5 methods:
4. AddParcelToFront(Parcel parcel) : This method will create a new Node with the parcel object as its data value and then add the newly created node to the front of the linked list.
5. DisplayInventory(): This method will traverse the linked list from its first node to its last node and print the data value (i.e., the id, customerName and weight of the parcel object) of each note.
6. RemoveParcelAtPosition(int n): This method will remove the node at position n in the linked list. Assume that the first node of the linked list has a position number of 1 and the second node has a position number of 2 and so on and so forth.

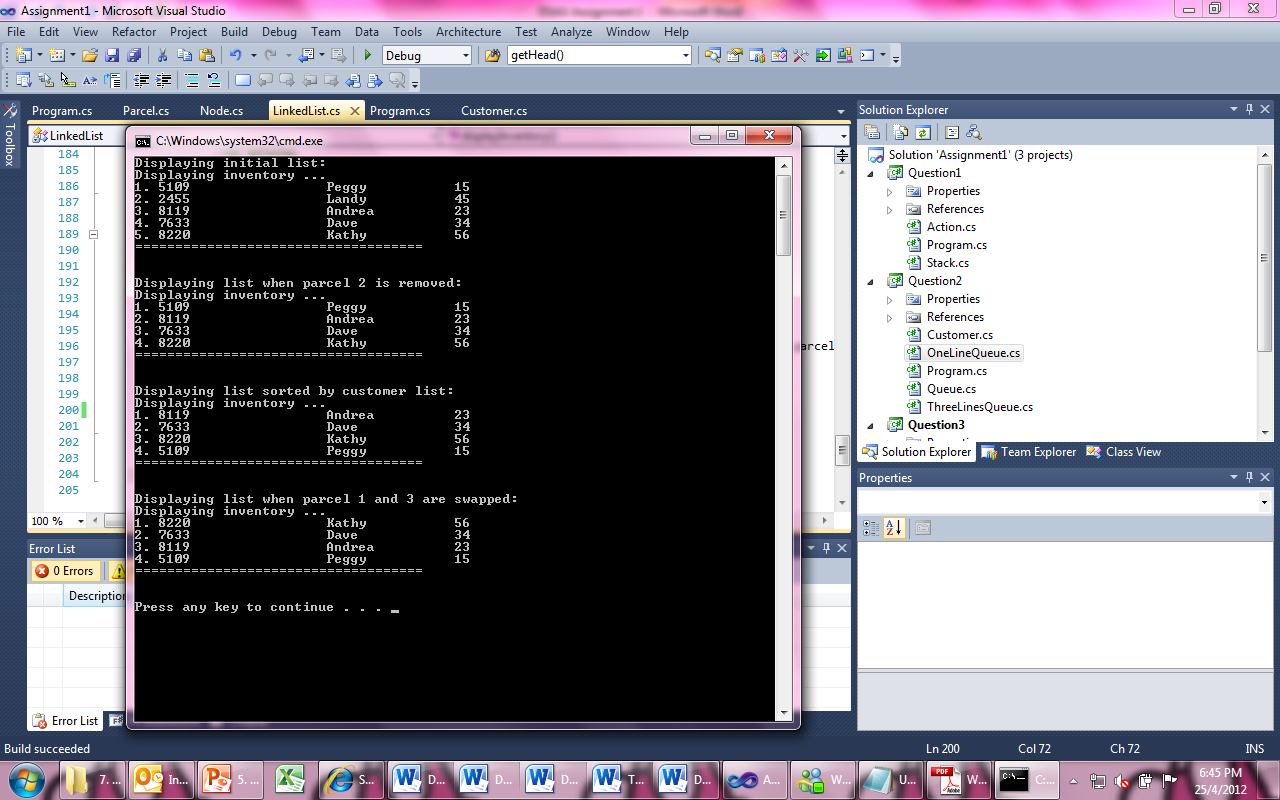
(stop at seven node means stop at 6 node link to node8)

1. ParcelSwap(int n1, int n2): This method will swap the node at position n1 with the node at position n2.
2. SortByCustomerName(): This method will sort the linked list by the customer name in ascending order.

**Notes:**

* You are allowed to make changes to the LinkedList.cs and Node.cs.
* You are not allowed to use Array for this question.
* You are not allowed to use any existing C# libraries to do sorting.
* Your solution should use objects and classes effectively.
* You are to write your own main program to test your solutions.

***The expected output when you run the program:***



**Status Report**

The status report must be typed-written using Microsoft Word with 1 inch margin all round, portrait, Times New Roman font, size 12, single line spacing. Proper English MUST be used for the writing of report.

Write a short summary of the final state of your assignment in a file called Status\_<YourName>\_<AdminNumber>.doc.

E.g. *Status\_LimChuKang\_0809115J.doc.*

For each program, state the following:

* Any assumptions made.
* Any question you did not complete with reasons for not completing the questions.
* Any problems you encountered while doing the assignment and how you go about resolving them.
* Any improvements / additional features attempted.

###### **Penalty for Late Submission Without Valid Reasons**

late and <1 day : 10% deduction from absolute mark given for the assignment

late>=1 and <2 days : 20% deduction from absolute mark

late>=2 days : No marks awarded

Note that “day” includes **non-working days** (Sat, Sun and public holidays).

General MC/LOA is NOT considered as valid reason for extended assignment submission.

**Grading Criteria**

The table below summarizes the grading criteria we use:

|  |  |
| --- | --- |
| **Grades** | **Criteria** |
| **F** | 1. Cannot complete questions 1 to 3b 2. Program failed to compile or run successfully and student did not attempt to seek help from tutor 3. Incomplete report   OR   1. **Clear evidence of plagiarism detected** (*Disciplinary actions will be taken against those who do so*) |
| **D / D+** | 1. Able to complete questions 1 to 3c(ii) 2. Program lack testing and robustness. Many (> 5) errors discovered during testing. 3. Mediocre but clear reporting 4. Presentation of output is not clear 5. Type-written report with much spelling and grammar errors |
| **C / C+** | 1. Able to complete questions 1 to 3c(iii) 2. Clear report with some thoughts of learning shown 3. Type-written report with some spelling and grammar errors 4. Presentation of output is clear 5. Some evidence of testing and robustness observed. Some (3-5) errors discovered. |
| **B / B+** | 1. Able to complete questions 1 to 3c(iv) 2. Clear report with well-thought reflections and learning shown 3. Type-written report with NO spelling and grammar errors 4. Presentation of output is clear. 5. Sufficient and clear comments in your codes 6. Program is well tested and robust, very few (<3) errors discovered. |
| **A / Z** | 1. Able to complete ALL questions 2. Initiate additional features (eg. added new functionality) 3. Clear report with well-thought reflections and learning shown 4. Type-written report with NO spelling and grammar errors 5. Presentation of output is clear. 6. Sufficient and clear comments in your codes 7. Program is well tested and robust, no errors discovered. |

\*\*\*\*\*End of Assignment\*\*\*\*\*