

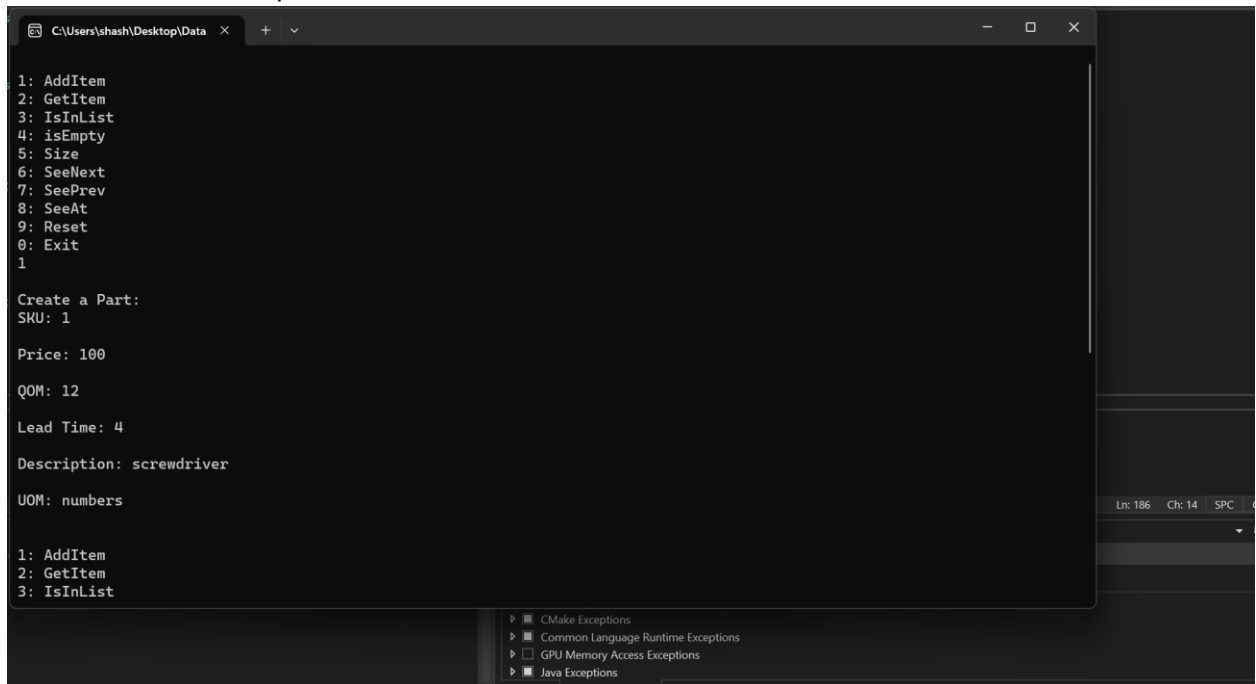
# Lab Report 8 : Doubly Linked Lists

## 1. Objectives and concepts:

The concepts explored in this lab included double linked lists, classes, constructors destructors, exception handling, etc. These concepts are important because linked list is a data structure that is very efficient as compared to simple arrays. Double linked list allows moving both forward and backward through the list.

These concepts are crucial for having a career in computer science as these are the fundamental data structures we are supposed to know and use when working in the industry.

## 2. Screenshots of the output:



```
C:\Users\shash\Desktop\Data x + v
1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
1

Create a Part:
SKU: 1

Price: 100

QOM: 12

Lead Time: 4

Description: screwdriver

UOM: numbers

1: AddItem
2: GetItem
3: IsInList
```

Debug - 32-bit - Continue - [35040] Main Thread - Stack Frame: Part::GetPartInfo

Exceptions.h Node.cpp Node.h Part.cpp Part.h lab8.cpp

(Global Scope) main()

```
catch (EndOfList)
{
    ex.what();
}
catch (EmptyList)
{
    ex.what();
}
break;
case 9: // reset
    List->Reset();
    break;
}

if (ans != 0)
{
    showMenu(); //
    cin >> MenuAns;
}

// te p;
// rn 0;

// s found

1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
2

Enter the SKU number of the part: 1
SKU: 1 Desc: screwdriver
```

Ln: 186 Ch: 14 SPC

Exceptions.h Node.cpp Node.h Part.cpp Part.h lab8.cpp

Exceptions

- CMake Exceptions
- Common Language Runtime Exceptions
- GPU Memory Access Exceptions
- Java Exceptions

Exceptions.h Node.cpp Node.h Part.cpp Part.h lab8.cpp

LinkedList<T> IsInList(T \* item)

```
1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
3

Enter the SKU number of the part: 1
part exists

1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
4

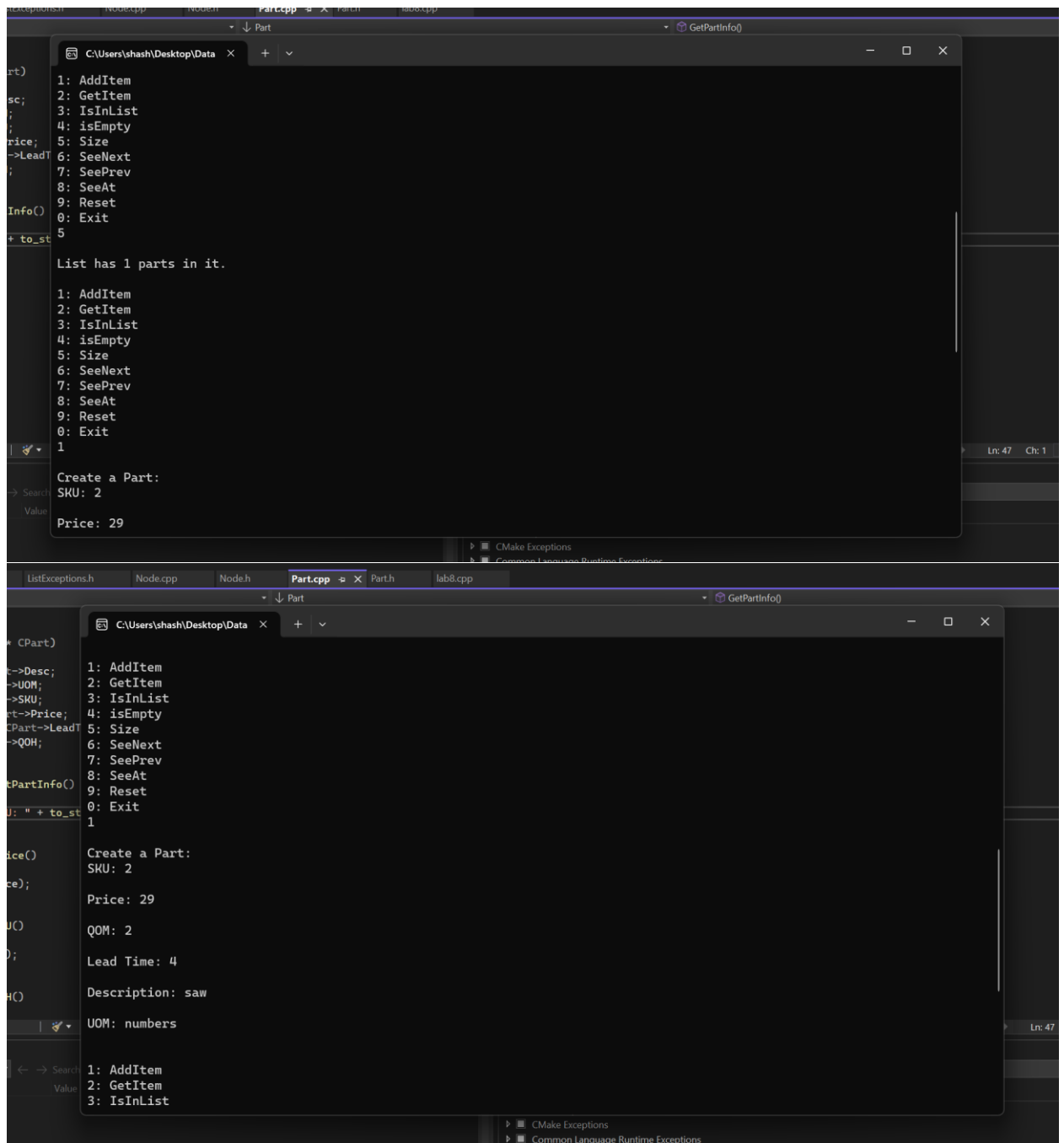
list not empty.
```

Ln: 142

0x0000000000000000 data=ffff, next=ffff, prev=ffff Node<Part> \*  
0x000001f2e6913aa0 [head=0x0000000000000000 <NULL> length=0 cu... LinkedList<Part> \*

Exceptions

- CMake Exceptions
- Common Language Runtime Exceptions



```
Part.cpp  Part.h  lab8.cpp
C:\Users\shash\Desktop\Data
1
Create a Part:
SKU: 2
Price: 29
QOM: 2
Lead Time: 4
Description: saw
UOM: numbers

1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
5

List has 2 parts in it.
```

```
Part.cpp  Part.h  lab8.cpp
C:\Users\shash\Desktop\Data
0: Exit
5

List has 2 parts in it.

1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
6

SKU: 2 Desc: saw

1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
```

```
listExceptions.h Node.cpp Node.h Part.cpp Part.h lab8.cpp
C:\Users\shash\Desktop\Data
0: Exit
6
SKU: 2 Desc: saw
1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
7
SKU: 1 Desc: screwdriver
1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
7
CMake Exceptions
Common Language Runtime Exceptions
```

```
exceptions.h Node.cpp Node.h Part.cpp Part.h lab8.cpp
C:\Users\shash\Desktop\Data
1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
8
Enter index in the list2
1
SKU: 2 Desc: saw
1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
8
Enter index in the list2
Ln: 47
CMake Exceptions
Common Language Runtime Exceptions
GPU Memory Access Exceptions
```

```
Part.cpp
1
SKU: 2 Desc: saw
1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
8
Enter index in the list2
0
SKU: 1 Desc: screwdriver
1: AddItem
2: GetItem
3: IsInList
4: isEmpty
5: Size
6: SeeNext
7: SeePrev
8: SeeAt
9: Reset
0: Exit
```