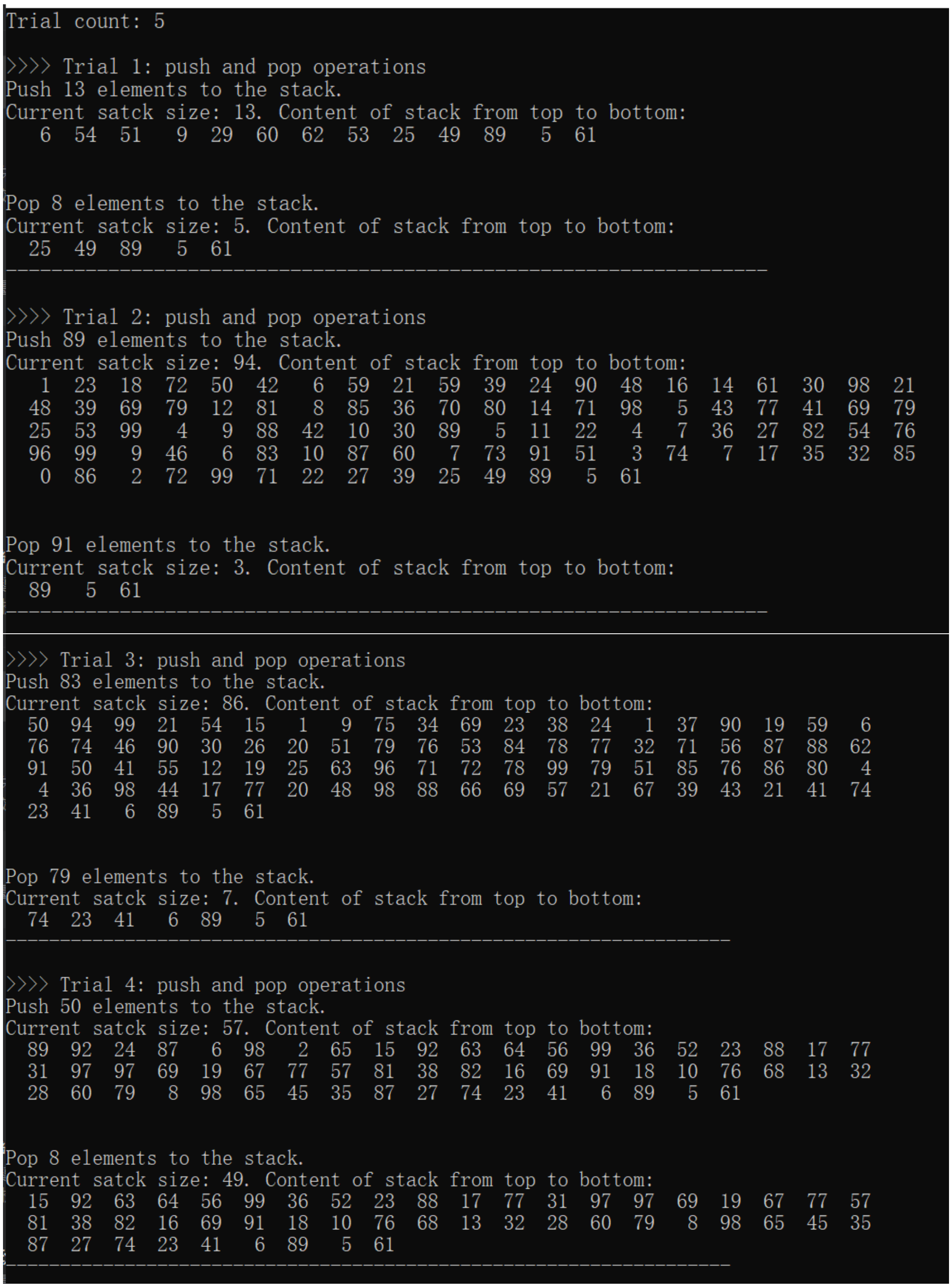
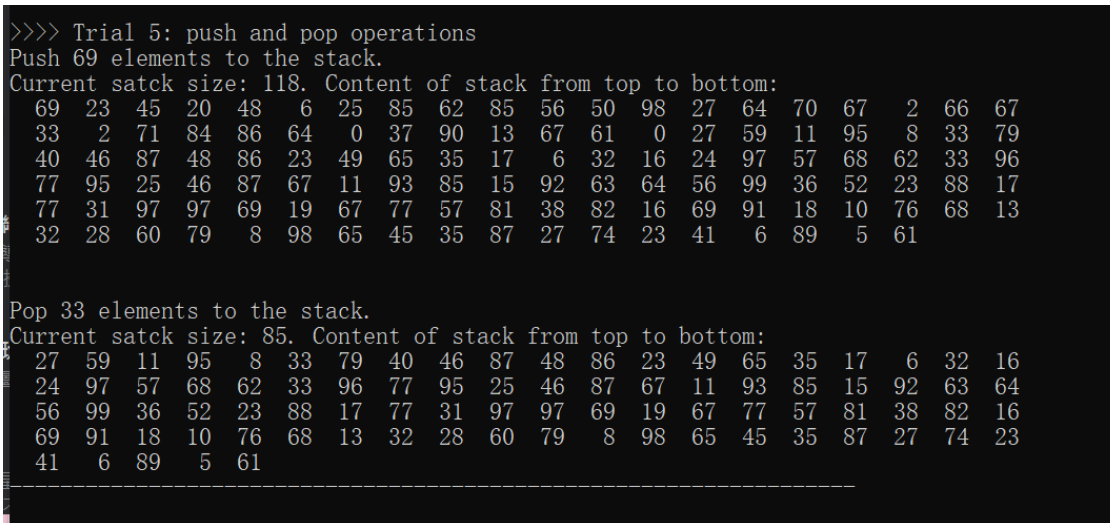
Programming Practice: Stacks Using Single-Linked Linear Lists

1. Define and implement a C++ project of an integer stack using single-linked linear list. Let the head of list be the top of a stack. In the application program, use random generator to get a trial count between 1 and 10. In each trial, randomly generate a number of push operations and a number of pop operations, and perform push and pop operation. Assume in each trial the number of pop operations is less than the current stack size. Solution: stack_single_list_head_as_top.rar. Example of program execution:



```
Trial count: 5

>>>> Trial 1: push and pop operations
Push 13 elements to the stack.
Current satck size: 13. Content of stack from top to bottom:
   6  54  51   9  29  60  62  53  25  49  89   5  61


Pop 8 elements to the stack.
Current satck size: 5. Content of stack from top to bottom:
  25  49  89   5  61
-------------------------------------------------------------------

>>>> Trial 2: push and pop operations
Push 89 elements to the stack.
Current satck size: 94. Content of stack from top to bottom:
   1  23  18  72  50  42   6  59  21  59  39  24  90  48  16  14  61  30  98  21
  48  39  69  79  12  81   8  85  36  70  80  14  71  98   5  43  77  41  69  79
  25  53  99   4   9  88  42  10  30  89   5  11  22   4   7  36  27  82  54  76
  96  99   9  46   6  83  10  87  60   7  73  91  51   3  74   7  17  35  32  85
   0  86   2  72  99  71  22  27  39  25  49  89   5  61


Pop 91 elements to the stack.
Current satck size: 3. Content of stack from top to bottom:
  89   5  61
-------------------------------------------------------------------

>>>> Trial 3: push and pop operations
Push 83 elements to the stack.
Current satck size: 86. Content of stack from top to bottom:
  50  94  99  21  54  15   1   9  75  34  69  23  38  24   1  37  90  19  59   6
  76  74  46  90  30  26  20  51  79  76  53  84  78  77  32  71  56  87  88  62
  91  50  41  55  12  19  25  63  96  71  72  78  99  79  51  85  76  86  80   4
   4  36  98  44  17  77  20  48  98  88  66  69  57  21  67  39  43  21  41  74
  23  41   6  89   5  61


Pop 79 elements to the stack.
Current satck size: 7. Content of stack from top to bottom:
  74  23  41   6  89   5  61
-------------------------------------------------------------------

>>>> Trial 4: push and pop operations
Push 50 elements to the stack.
Current satck size: 57. Content of stack from top to bottom:
  89  92  24  87   6  98   2  65  15  92  63  64  56  99  36  52  23  88  17  77
  31  97  97  69  19  67  77  57  81  38  82  16  69  91  18  10  76  68  13  32
  28  60  79   8  98  65  45  35  87  27  74  23  41   6  89   5  61


Pop 8 elements to the stack.
Current satck size: 49. Content of stack from top to bottom:
  15  92  63  64  56  99  36  52  23  88  17  77  31  97  97  69  19  67  77  57
  81  38  82  16  69  91  18  10  76  68  13  32  28  60  79   8  98  65  45  35
  87  27  74  23  41   6  89   5  61
-------------------------------------------------------------------
```



```
>>>> Trial 5: push and pop operations
Push 69 elements to the stack.
Current satck size: 118. Content of stack from top to bottom:
  69  23  45  20  48   6  25  85  62  85  56  50  98  27  64  70  67   2  66  67
  33   2  71  84  86  64   0  37  90  13  67  61   0  27  59  11  95   8  33  79
  40  46  87  48  86  23  49  65  35  17   6  32  16  24  97  57  68  62  33  96
  77  95  25  46  87  67  11  93  85  15  92  63  64  56  99  36  52  23  88  17
  77  31  97  97  69  19  67  77  57  81  38  82  16  69  91  18  10  76  68  13
  32  28  60  79   8  98  65  45  35  87  27  74  23  41   6  89   5  61


Pop 33 elements to the stack.
Current satck size: 85. Content of stack from top to bottom:
  27  59  11  95   8  33  79  40  46  87  48  86  23  49  65  35  17   6  32  16
  24  97  57  68  62  33  96  77  95  25  46  87  67  11  93  85  15  92  63  64
  56  99  36  52  23  88  17  77  31  97  97  69  19  67  77  57  81  38  82  16
  69  91  18  10  76  68  13  32  28  60  79   8  98  65  45  35  87  27  74  23
  41   6  89   5  61
--------------------------------------------------------------------
```

2. Repeat Question 1with that change of let the head of list be the bottom of a stack. (no solution provided)