**Short Answers**

1. I am going to execute this code with THREE pushes and ONE pop:

stack<int> s;

s.push(1);

s.push(2);

s.push(3);

s.pop( );

Suppose that s is represented by a partially filled array. Draw the state of the private member variables of s after the above code: used is 2 see picture below \*\*\*\*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | 2 | 3 |  |  |

Data

[0] [1] [2] [3] [4]

\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

used| | data| | | | | |

|\_\_\_\_\_\_\_| |\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_|\_\_\_\_\_\_|

[0] [1] [2] [3] [4]

1. I am going to execute this code with THREE pushes and ONE pop:

stack<int> s;

s.push(1);

s.push(2);

s.push(3);

cout << s.pop( );

Suppose that s is represented by a linked list. Draw the state of the private member variables of s after the above code: Head is 2 then 1

\_\_\_\_\_\_\_

head\_ptr| |

|\_\_\_\_\_\_\_|

**Multiple Choice**

1. Entries in a stack are "ordered". What is the meaning of this statement?
   * A. A collection of stacks can be sorted.
   * B. Stack entries may be compared with the '<' operation.
   * C. The entries must be stored in a linked list.
   * D. There is a first entry, a second entry, and so on.
2. The operation for adding an entry to a stack is traditionally called:
   * A. add
   * B. append
   * C. insert
   * D. push
3. The operation for removing an entry from a stack is traditionally called:
   * A. delete
   * B. peek
   * C. pop
   * D. remove
4. Which of the following stack operations could result in stack underflow?
   * A. is\_empty
   * B. pop
   * C. push
   * D. Two or more of the above answers
5. Which of the following applications may use a stack?
   * A. A parentheses balancing program.
   * B. Keeping track of local variables at run time.
   * C. Syntax analyzer for a compiler.
   * D. All of the above.
6. Consider the following pseudocode:

declare a stack of characters

while ( there are more characters in the word to read )

{

read a character

push the character on the stack

}

while ( the stack is not empty )

{

write the stack's top character to the screen

pop a character off the stack

}

What is written to the screen for the input "carpets"?

* + A. serc
  + B. carpets
  + C. steprac
  + D. ccaarrppeettss

1. Here is an INCORRECT pseudocode for the algorithm which is supposed to determine whether a sequence of parentheses is balanced:

declare a character stack

while ( more input is available)

{

read a character

if ( the character is a '(' )

push it on the stack

else if ( the character is a ')' and the stack is not empty )

pop a character off the stack

else

print "unbalanced" and exit

}

print "balanced"

Which of these unbalanced sequences does the above code think is balanced?

* + A. ((())
  + B. ())(()
  + C. (()()))
  + D. (()))()

1. Suppose we have an array implementation of the stack class, with ten items in the stack stored at data[0] through data[9]. The CAPACITY is 42. Where does the push member function place the new entry in the array?
   * A. data[0]
   * B. data[1]
   * C. data[9]
   * D. data[10]
2. In the linked list implementation of the stack class, where does the push member function place the new entry on the linked list?
   * A. At the head
   * B. At the tail
   * C. After all other entries that are greater than the new entry.
   * D. After all other entries that are smaller than the new entry.