Stack and Queues

* Required

If the characters 'D', 'C', 'B', 'A' are placed in a queue (in that order), and then removed one at a time, in what order will they be removed? * 1 point

- ABDC
- ABCD
- DCAB
- DCBA

If the characters 'D', 'C', 'B', 'A' are placed in a stack (in that order), and then removed one at a time, in what order will they be removed? * 1 point

- ABDC
- DCBA
- DCAB
- ABCD

Predict the output of code * 2 points

assume no compilation error

```
int values[] =  {1, 3, 5, 7, 9, 11, 13, 15, 17, 19 };
Stack s;  //consider stack of integers
for (int 1 = 0; i < 10; i++)
        s.push( values[ i ] );
int n = 25;
for (int i = 0; i < 4; i++)
{
        n += s.pop( );
}
for (int i = 0; i < 2; i++)
{
        n -= s.pop( );
}
printf( n );
```

69

What fun() function does? * 2 points

```
void fun(int n)
{
   Queue q;
   enqueue(&q, 0);
   enqueue(&q, 1);
   for (int i = 0; i < n; i++)
   {
      int a = dequeue(&q);
      int b = dequeue(&q);
      enqueue(&q, b);
      enqueue(&q, a+b);
      print(a);
   }
}
```

0

Following is a C like pseudo-code for a function that takes a Queue as an argument, and uses a stack S to do the processing. What does the function do in general? * 4 points

```
void fun(Queue *Q)
{
   Stack S;   // Say it creates an empty stack S

   // Run while Q is not empty
   while (!isEmpty(Q))
   {
      // deQueue an item from Q and push the dequeued item to S
      push(&S, deQueue(Q));
   }

   // Run while Stack S is not empty
   while (!isEmpty(&S))
   {
      // Pop an item from S and enqueue the poppped item to Q
      enQueue(Q, pop(&S));
   }
}
```

Reversing the order of queue

Submit