**Classwork 10.1(2) grade**  Given a score $x$, a mean $m$ and a std. dev. $s$, $\text{grade}(x,m,s)$ gives the score’s letter grade which is

- $A$ if $x$ is 2 std. dev. above the mean,
- $B$ if $x$ is between 1 std. dev. & 2 std. dev. above the mean,
- $C$ if $x$ is between 1 std. dev. below & 1 std. dev. above,
- $D$ if $x$ is between 1 std. dev. & 2 std. dev. below the mean,
- $F$ if $x$ is 2 std. dev. below the mean,

Resolve $< \text{ vs. } \leq$ in favor of the student.

```
//c10.1(2)grade  mode(0);warning('off');printf('\n');
function g=grade(x,m,s)
if(m+2*s<=x) then; g='A'
elseif(m+s<=x & x<m+2*s) then; g='B'
end
endfunction
m=50; s=13;
for i=80:-5:20
    printf(" %i %s\n", i,grade(i,m,s))
end
```
**Homework 10.1(3)** *grade2* Write a function *grade2*(n) with integer input which returns

A if $9 \leq n$
B if $7 \leq n < 9$
C if $3 < n < 7$
D if $1 < n \leq 3$
F if $n \leq 1$

The output should be as follows:

<table>
<thead>
<tr>
<th>n</th>
<th>grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>A</td>
</tr>
<tr>
<td>9</td>
<td>A</td>
</tr>
<tr>
<td>8</td>
<td>B</td>
</tr>
<tr>
<td>7</td>
<td>B</td>
</tr>
<tr>
<td>6</td>
<td>C</td>
</tr>
<tr>
<td>5</td>
<td>C</td>
</tr>
<tr>
<td>4</td>
<td>C</td>
</tr>
<tr>
<td>3</td>
<td>D</td>
</tr>
<tr>
<td>2</td>
<td>D</td>
</tr>
<tr>
<td>1</td>
<td>F</td>
</tr>
<tr>
<td>0</td>
<td>F</td>
</tr>
</tbody>
</table>

```c
//h10.1(3)grade2
mode(0);warning('off');printf('
');

...delete this line, write the function
for n=10:-1:0
    printf(" %i %s\n", n, grade2(n))
end
```
**Classwork 10.2(2)** reverse Let list be a vector of n integers. Write a function reverse(list) which returns the result of listing the elements of list in the reverse order. If list=[4,2,3,1], then reverse(list)=[1,3,2,4].

```c
//c10.2(3) reverse
mode(0);warning('off');printf('
');
...delete this line, write the function
a=[1 2 3], reverse(a), a=[5 4 3 2], reverse(a)
```
Classwork 10.3(3) insert  Let list be a vector of integers. Write a function insert(list, k, x) which, given k in 1:n inserts the number x into position k and pushes the existing records from k to n down to positions k+1 and n+1.

Inserting a x=9 at position k=2 in the list on the left, should give the list on the right.

| list(1)= 0 | list(1)= 0 |
| list(2)= 1 | list(2)= 9 |
| list(3)= 2 | list(3)= 1 |
| list(4)= 3 | list(4)= 2 |
| list(5)= 4 | list(5)= 3 |
| list(6)= 4 | list(6)= 4 |

//c10.3(3)insert
mode(0); warning('off'); printf('
');

...delete this line, write the function
list=1:5
for k=1:6; printf('\nInsert 10 at position %i',k)
disp(list); disp(insert(list,k,10)); end;
Homework 10.2(4) delete  Let list be a list of n integers. Write a function delete(list, k) which, given k in 1:n deletes the kth element of list and moves any elements below the kth position up to any gap in the list to give a vector of length n-1.

Deleting the element in position k=2 element from the list on the left, gives the shorter list on the right.

| list(1) = 0 | list(1) = 0 |
| list(2) = 1 | list(2) = 2 |
| list(3) = 2 | list(3) = 3 |
| list(4) = 3 | list(4) = 4 |
| list(5) = 4 |

//h10.2(4)delete
mode(0);warning('off');printf('
');

...delete this line, write the function
list=1:5;
for k=1:5 ; printf("\nDelete at position %i ",k);
    disp(list);disp(delete(list,k));
end;
Classwork 10.4(3)  find_first  Let list be a list of n integers and let i be an integer which may or may not be in the list. Write a function find_first(list, x) which finds the first position of x in the list. If x is not in the list, return 0.

find([7,7,5,5],1)=0
find([7,7,5,5],7)=1
find([7,7,5,5],5)=3

//c10.4(3) find

mode(0);warning('off');printf('
');

...delete this line, write the function

list=[5 5 6 6 7 7]
for k=4:8
    printf("Item sought=%i",k)
    printf(" First position found=%i \n",find(list,k))
end
**Homework 10.3(3)**  

**find_last**  
Let list be a list of n integers and let i be an integer which may or may not be in the list. Write a function `find_last(list, x)` which finds the last position of x in the list. If x is not in the list, return 0.

`find_last([7,7,6,5],1)=0`  
`find_last([7,7,6,5],7)=2`  
`find_last([7,7,5,5],5)=4`

```matlab
//h10.3(3) find_last
mode(0);warning('off');printf('
');

...delete this line, write the function
list=[5 5 6 6 7 7]
for k=4:8
  printf("Item sought %i",k)
  printf(" Last position %i \n",find_last(list,k))
end
```
**Classwork due at end of period**
email to: dale@math.hawaii.edu  subject line: 190 c10(10) 10.1(2) grade, 10.2(2) reverse, 10.3(3) insert, 10.4(3) find

**Homework due next class**
email to: dale@math.hawaii.edu  subject line: 190 h10(10) 10.1(3) grade2, 10.2(4) delete, 10.3(3) find_last


*This next homework is the first lecture after the midterm.**

**Homework: Creativity Test h11.0(6) swap.** Replace the two red lines with three lines of code which swap the values of x and y. Due first class after the two midterms.

//h11.0(6)

//Write three lines which swap the values of x and y.
mode(0);warning('off');printf('\n');
x=input('Enter x >');
y=input('Enter y >');
printf('%i %i
',x,y)
\begin{verbatim}
x=y;
y=x;
printf('%i %i\n',x,y)
\end{verbatim}

email: dale@math.hawaii.edu  subject line: 190 h11.0(6)

Due first class after the two midterms.

We want the two red lines to swap the values of $x$ and $y$. Hence if $x=2; y=3$, then after executing the code in you should get $x=3; y=2$. Running the code shows that two red lines do not swap the values of $x$ and $y$. Write code which does work. You can use variables other than $x$ and $y$.

Don’t exchange $x$ and $y$ in the last printf line. This changes the print order but does not change the values of $x$ and $y$.

This program is the simplest problem which requires original thinking. If you are not creative, it will be hard to solve it or any other programming problem requiring originality. If you are not creative, it is hard to write a program without an example to work with. If in previous classes you needed worked examples to solve a problems, you may have trouble doing original thinking. You should be able to do the closed-book problems which are similar to previously problems. But you may have difficulty with open-book problems don't have examples to copy or modify.