```
1   #include<stdio.h>
2   #include<conio.h>
3   #include<ctype.h>
4
5   #pragma warning(disable:4996)
6
7   void v_or_c(int iLetter);
8   /*------------------------------------------------------------------------*/
9   void main(void)
10  {
11    int iLetter;
12
13    do
14    {
15      printf("\nEnter a letter.  Enter Q to quit. => ");
16      iLetter = getch();
17      printf("%c",iLetter);        /*use getche() to make this unnecessary*/
18      iLetter = toupper(iLetter);
19      v_or_c(iLetter);
20    }
21    while (iLetter != 'Q');
22  }
23  /*------------------------------------------------------------------------*/
24  void v_or_c(int iLetter)
25  {
26    if ((iLetter >= 'A') && (iLetter <='Z'))
27    {
28      switch (iLetter)
29      {
30        case 'A' :
31        case 'E' :
32        case 'I' :
33        case 'O' :
34        case 'U' : printf("\nYou chose a vowel.");
35        break;
36        case 'Q' :
37        break;
38        default  : printf("\nYou choose a consonant");
39        break;
40      }
41    }
42    else
43      printf("\nYou did not chose a Letter.");
44  }
45
46  /*
47  Enter a letter.  Enter Q to quit. => a
48  You chose a vowel.
49  Enter a letter.  Enter Q to quit. => b
50  You chose a consonant
51  Enter a letter.  Enter Q to quit. => 3
52  You did not choose a Letter.
53  Enter a letter.  Enter Q to quit. => q
54  Press any key to continue . . .
55  */
56
```

Commented [KW1]: Gets a single character without having to hit the enter key so no \n character is stored in the input data stream.

Commented [KW2]: toupper makes lower case characters into upper case.

Commented [KW3]: The do-while loop is used in this instance to keep the program running until a Q or q is selected. It prompts the user each time it loops.

Commented [KW4]: Test to make sure the alphabet character is within bounds. If it is, use the switch case construct to test to see if it is a vowel or consonant.

Commented [KW5]: If any of these cases are true, the cases below the true case will execute as well until a break is found.

Commented [KW6]: Lets the program user know they did not follow directions.

Commented [KW7]: All test conditions tested until the program encountered a q.

```
#include<stdio.h>
#include<conio.h>

#pragma warning(disable : 4996)

void print_5_cubes(float fVal);
/*---------------------------------------------------------------------*/
void main (void)
{
  float fIn_val;

  printf("Enter a starting number to cube => ");
  scanf("%f",&fIn_val);
  print_5_cubes(fIn_val);

}
/*---------------------------------------------------------------------*/
void print_5_cubes(float fVal)
{
  float fVal_cubed;
  int iCount;

  printf("\n   number        cube \n \n");

  for (iCount = 1; iCount <= 5; ++iCount, fVal += 2)
  {
    fVal_cubed = fVal * fVal * fVal;
    printf("%8.2f     %8.2f \n",fVal,fVal_cubed);
  }
}
/*
Enter a starting number to cube => 2

   number        cube

    2.00         8.00
    4.00        64.00
    6.00       216.00
    8.00       512.00
   10.00      1000.00

 Push a key to return to editor.
*/
```

Commented [KW8]: Two variables are updated in this field by using the comma operator.

Commented [KW9]: The for loop is used create a table based on the boundary conditions in the test field.

Commented [KW10]: Result of the program output when a 2 is entered.

```c
#include<stdio.h>
#include<conio.h>
#include<ctype.h>

#pragma warning(disable:4996)

int make_sel(int iN1, int iN2);
void cal_dsply_sum(int iN1);
void cal_dsply_product(int ifN2);
/*-------------------------------------------------------------------------*/
int main(void)
{
  int iSelect, iN1, iN2;

  printf("\nEnter Max number for sum  => ");
  scanf("%d",&iN1);
  while (getchar() != '\n');
  printf("Enter Max Number for product => ");
  scanf("%d",&iN2);
  while (getchar() != '\n');

  do
  {
            iSelect = make_sel(iN1, iN2);
            switch (iSelect)
            {
              case 'A': cal_dsply_sum(iN1);
              break;
              case 'B': cal_dsply_product(iN2);
              break;
              case 'Q': break;
              default : printf("\n\nYou did not enter A, B, or Q \n");
            }
  }
  while (iSelect != 'Q');
  return 0;
}
/*-------------------------------------------------------------------------*/
int make_sel(int iN1, int iN2)
{
  int cChoice;

  printf("\n\nA - calculate sum from 0 to %d \n",iN1);
  printf("B - calculate product from 1 to %d \n", iN2);
  printf("Q - quit");
  printf("\nSelect A, B, or Q => ");
  cChoice =(toupper (getche()));
  puts("\n");
  return cChoice;
}
/*-------------------------------------------------------------------------*/
void cal_dsply_sum(int iN1)
{
  int sum = 0, index = 1;

  while (index <= iN1)
  {
            sum = sum + index   ;
            ++index;
  }
  printf("Sum of integers from 0 to %d is %d \n", iN1, sum);
}
/*-------------------------------------------------------------------------*/
```

Commented [KW11]: scanf is used to enter a character and stored as an integer. The scanf always appends a \n, so this character creates a testing issue when its stored in the input stream

Commented [KW12]: This while loop runs until it clears the \n character out of the input stream. It allows the scanf to be used in this application where the program is testing for a single character or integer.

Commented [KW13]: iSelect is updated before the switch case to ensure the loop operates correctly. If iSelect is not updated correctly, the loop may never terminate or maybe even run.

Commented [KW14]: Get character with echo which means it is displayed on the screen.

Commented [KW15]: puts prints a string without format specifiers. However, the \n character can be added since it is just a control character.

Commented [KW16]: Must initial before entering loop.

Commented [KW17]: while loop used to sum up a series of integers. Notice the variable sum is initialized to 0 before entering the loop and index is initialized to 1.

```
void cal_dsply_product(int iN2)
{
  int product = 1, index = 1;
  do{
             product = product * index;
             ++index;
    }
  while (index <= iN2);
  printf("%d factorial = %d \n", iN2, product);
}
/*-------------------------------------------------------------------------*/
 /*
Enter Max number for sum  => 5
Enter Max Number for product => 7


A - calculate sum from 0 to 5
B - calculate product from 1 to 7
Q - quit
Select A, B, or Q => a

Sum of integers from 0 to 5 is 15


A - calculate sum from 0 to 5
B - calculate product from 1 to 7
Q - quit
Select A, B, or Q => b

7 factorial = 5040


A - calculate sum from 0 to 5
B - calculate product from 1 to 7
Q - quit
Select A, B, or Q => q
*/
```

Commented [KW18]: Must initialize before entering loop.

Commented [KW19]: do-while loop used to create a factorial result. In this case, index can be updated at the end of the loop since the do-while is guaranteed to run at least once.

Commented [KW20]: Program results.