**Question 1**

Write a program that requires users to key-in their tuition fee. The program will calculate the total fees for the month and the average payment to be made. Create a dynamic array with the *new* operator to store the elements entered by user.

**[HINT: Complete the given program by referring to the sample output]**

|  |
| --- |
| #include <iostream>  using namespace std;  int main ( )  { int subject, num;  float total\_payment, average\_payment;  float \*fees;    /\* Ask user for the number of subjects registered \*/  ----------  ----------    fees= new float[subject];  for (num=0; num<subject; num++)  /\* Ask user for the fees charged for each class registered. Then, calculate the total payment \*/  ----------  ----------  ----------  ----------  /\* Display the details of the fees entered earlier \*/  ----------  ----------  /\* Calculate the average payment, then display the total payment & minimum payment (which is the average).\*/  ----------  ----------  ----------  ----------    return 0;  } |

|  |
| --- |
| **Sample Output Screen** |
| How many subjects you have registered: 3  Class 1 fees charge: RM 60  Class 2 fees charge: RM 50  Class 3 fees charge: Rm 70  ===== PAYMENT DETAILS =====  Fees charges you have entered: 60(RM)... 50(RM)...70(RM)...  Total payment to be made: RM 180  Minimum payment (Average of total payment): RM 60 |

**Question 2**

Write a C++ program that contains:

1. A constant global variable ***ITEM*** with value **3**.
2. A class ***BeanBag*** with the following:
   1. Private data members: **code** ***name* (string); *stock [ITEM]* (int)**
   2. Public member functions:
      * **latest\_BeanBagStock ( )**
        + - Display the ***“#Current# Ready Stocks…….”*** with the code *name* and the array elements of *stock* in reverse order using an appropriate looping structure.
      * **beanBag\_Details (int \* )**
        + - This function prints details of ***“Stocks Checking”***
          - Gets user input for the code *name.*
          - The function has a pointer argument.
          - In a *for* loop, use the pointer argument to initialize the *stock* array.
      * A global object declaration, named ***ready*.**
3. A function named **stockUpdate()**:
   1. Refer to label ‘***stockUpdate******( )***’ at sample output.
   2. Get user input for 3 values that should be stored in a dynamic array created with the *new* operator.
   3. Using the global object ***ready***, call the function ***beanBag\_Details(…)***, passing in the array and also call ***latest\_BeanBagStock()*** after that.
   4. Delete the dynamic array created.
4. In the *main()*:
5. Declare an object of the class BeanBag
6. Declare an array of 3 integer elements and initialize it with the values {9,7,5}
7. Using the object (created at d)(i)), call ***beanBag\_Details (….)*** with the array declared at d)(ii).
8. Using the object (created at d)(i)), call ***latest\_BeanBagStock ()***.
9. Call ***stockUpdate()***

|  |
| --- |
| **Sample Output Screen** |
| COSY Bean Bag Chair Sdn. Bhd.  #######################################  stocks Checking  ---------------------------------------  Enter Bean Bag Chair code: AB102  --------------------------------------  #Current# Ready stocks Checking...  --------------------------------------  Code Tracing >>AB102<<  Group 1 production: 5 item(s) ready  Group 2 production: 7 item(s) ready  Group 3 production: 9 item(s) ready  ------------------------------------------  Ready stocks for this Month  ------------------------------------------  Ready stock from group 3:90  Ready stock from group 2:54  Ready stock from group 1:71  #######################################  stocks Checking  ---------------------------------------  ***stockUpdate******( )***  Enter Bean Bag Chair code: BQ103  --------------------------------------  #Current# Ready stocks Checking...  --------------------------------------  Code Tracing >>BQ103<<  Group 1 production: 71 item(s) ready  Group 2 production: 54 item(s) ready  Group 3 production: 90 item(s) ready  ------------------------------------------ |