**Question No. 01:**

Factor calculator is a program which asks user how many inputs he wants to take and then asks user to

enter n numbers such that, Numbers entered should be greater than 6 and less than 100. If the number

does not fall in this range, program should alert user to re-enter the number. If the entered number is 1, the program should stop taking input and return the factors of all the entered numbers. Create factors array for each number in above allocated 2D array and store that number on 1st index of factors array and the factors on the further indexes. Your program should also de-allocate acquired dynamic memory at the end of the life cycle. Do not consume extra memory.

Int\*\* GetFactorsArray(...) // This function takes count of elements and values of elements from user.

Returns a dynamic 2-D Array of Factors of the elements(numbers) entered.

Void Print(.......); Displays 2-D array in proper format.

Void main()

{

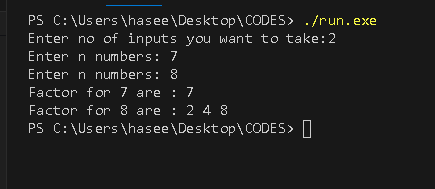
// Design main as per your requirements.

}

**Source Code**

|  |
| --- |
| #include <iostream>  using namespace std;  int \*GetFactorsArray(int n)  {      int \*factor = new int[n];      int count = 0;      for (int i = 1; i <= n; ++i)      {          if (n % i == 0)          {              factor[count++] = i;            }      }      factor[count] = -1;      // return factor column      return factor;  };  void print(int \*\*factorArray, int input)  {      for (int i = 0; i < input; i++)      {          cout << "Factor for " << factorArray[i][0] << " are : ";          // finite loop sentimaental value;          for (int j = 1; factorArray[i][j] !=-1; j++)          {              cout << factorArray[i][j] << " ";          }          cout << endl;      }  }  int main()  {      // int m = 3, n = 4, c = 0;      // for (int i = 0; i < m; i++)      // {      //     factor[i] = new int[n];      // }      int inputs;      cout << "Enter no of inputs you want to take:";      cin >> inputs;      int \*\*factor;      factor = new int \*[inputs];      for (int i = 0; i < inputs; i++)      {          int n;          cout << "Enter n numbers: ";          cin >> n;          while (n < 6 || n > 100)          {              cout << "Wrong input. Please re-enter the number: ";              cin >> n;          }          if (n == 1)          {              break;          }          factor[i] = GetFactorsArray(n);          factor[i][0] = n;          // factor[i] = GetFactorsArray(n);      }      // for (int i = 0; i < inputs; ++i)      // {      //     for (int j = 0; j < factor[i][0]; ++j)      //     {      //         cout << factor[i][j] << " ";      //     }      //     cout << endl;      // }      print(factor, inputs);      for (int i = 0; i < inputs; i++)      {          delete[] factor[i];      }      delete[] factor;      return 0;  } |

Output



// Sample Output:

// Enter no of inputs you want to take: 2

// Enter Number between 6 and 100: 12

// Enter Number between 6 and 100: 5

// Wrong input. Please re-enter the number

// Enter Number between 6 and 100:10

// ...

// Following are the factors of numbers entered:

// Factors of 12 are 1,2,3,4,6,12

// Factors of 10 are 1, 2,5,10

**Question No. 02:**

Define a class to represent a Bank account. Include the following members. Data members: -

• Name of the depositor

• Account number.

• Type of account.

• Balance amount in the account.

Provide a default constructor, a parameterized constructor and a copy constructor to this class.

Also provide Member Functions: -

1. To deposit an amount.

2. To withdraw the amount after checking for minimum balance.

3. To display all the details of an account holder.

Illustrate all the constructors as well as all the methods by defining objects. In the main function,

dynamically allocate an array of Bank Account objects. Your program should ask the user about

the size of the array and then allocate the array dynamically. Then, your program should ask the

user to enter the values of all attributes (Name of the depositor Account number, Type of

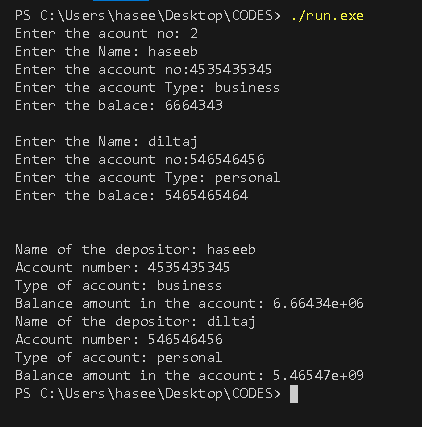
account, Balance amount in the account) for each Bank Account. After that, your program should

display the total number of Bank Account details as well as the details of each object on screen

(by calling the display function for each Bank Account Object).

|  |
| --- |
| #include <iostream>  #include <cstring>  using namespace std;  class Bank\_account  {      string Name\_of\_the\_depositor;      char Account\_number[16];      char Type\_of\_account[12];      long double Balance\_amount\_in\_the\_account;  public:      Bank\_account()      {          Balance\_amount\_in\_the\_account = 0;      }      Bank\_account(const string n,const char acc\_num[],const char type[],const double balance) : Name\_of\_the\_depositor(n)      {          Balance\_amount\_in\_the\_account = balance;          strncpy(Account\_number, acc\_num, sizeof(Account\_number));          Account\_number[sizeof(Account\_number) - 1] = '\0';          strncpy(Type\_of\_account, type, sizeof(Type\_of\_account));          Type\_of\_account[sizeof(Type\_of\_account) - 1] = '\0';      }      Bank\_account(const Bank\_account &obj)      {          Name\_of\_the\_depositor = obj.Name\_of\_the\_depositor;          strncpy(Account\_number, obj.Account\_number, sizeof(Account\_number));          Account\_number[sizeof(Account\_number) - 1] = '\0';          strncpy(Type\_of\_account, obj.Type\_of\_account, sizeof(Type\_of\_account));          Type\_of\_account[sizeof(Type\_of\_account) - 1] = '\0';          Balance\_amount\_in\_the\_account = obj.Balance\_amount\_in\_the\_account;      }      void Deposit()      {          cout << "Enter the Amount you want to deposite: ";          double deposite;          cin >> deposite;          Balance\_amount\_in\_the\_account += deposite;      }      void Withdraw()      {          if (Balance\_amount\_in\_the\_account == 0)          {              cout << "Insufficent balance." << endl;          }          else          {              cout << "Enter the Withdraw Amount ";              double amount;              cin >> amount;              if (amount > Balance\_amount\_in\_the\_account)              {                  cout << "Insufficent balance." << endl;              }              else              {                  Balance\_amount\_in\_the\_account -= amount;                  cout << "Withdrawal succesfully. Remaining balance is: " << Balance\_amount\_in\_the\_account << endl;              }          }      }      void display()      {          cout << "Name of the depositor: " << Name\_of\_the\_depositor << endl;          cout << "Account number: " << Account\_number << endl;          cout << "Type of account: " << Type\_of\_account << endl;          cout << "Balance amount in the account: " << Balance\_amount\_in\_the\_account << endl;      }  };  int main()  {      cout << "Enter the acount no: ";      int size;      cin >> size;      Bank\_account\* cust = new Bank\_account[size];      for (int i = 0; i < size; i++)      {          char Type\_of\_account[12];          char Account\_number[16];          string name;          double balan;          cout << "Enter the Name: ";          cin.ignore();          getline(cin,name);          cout << "Enter the account no:";          cin >> Account\_number;          cout << "Enter the account Type: ";          cin >> Type\_of\_account;          cout << "Enter the balace: ";          cin >> balan;          // also possible throiugh samart pointer = unique pointer          cust[i] = Bank\_account(name, Account\_number, Type\_of\_account, balan );          cout << endl;      }      cout << endl;      for (int i = 0; i < size; i++)      {          cust[i].display();      }        delete[] cust;        return 0;  } |

**Output:**

****