Question 1

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

class Maximum

{

int m, z, sz;

public:

    //constructor

    int \*cop;

    Maximum(int \*a, int b)

    {

        cop = new int[b];

        for(int i=0; i<b; ++i)

            cop[i] = a[i];

        z = b; //need to initiate because 'b' is not initialized in public class globally

    }

    int GetMaximum();//function

};

int Maximum::GetMaximum()

{

    m = cop[0];

    sz = z;

    for(int i=0; i<sz; i++)

    {

        if(cop[i] > m)

        {

            m = cop[i];

        }

    }

    return m;

}

int main()

{

    int sz = 5;

    int arr[sz];

    srand((unsigned)time(0));

    for(int i=0; i<sz; i++)

    {

        arr[i] = rand()%10 + 1;

        cout << "array[" << i <<"] = " << arr[i] << endl;

    }

    Maximum GetMax(arr, sz); //access to constructor

    cout << "Maximum is " << GetMax.GetMaximum() << endl; //constructor link to function

    return 0;

}

Question 2

#include <iostream>

using namespace std;

class OddPrint

{

int low, hig, init;

public:

    OddPrint(int, int);

    void Print();

};

OddPrint::OddPrint(int a, int b)

{

    low = a;

    hig = b;

}

void OddPrint::Print()

{

    if(low%2 == 1)

        init = low;

    else

        init = low + 1;

    for(int i=init; i<=hig; i++)

    {

        if(i%2 == 1)

        {

            cout << i << endl;

        }

        else

        {

            continue;

        }

    }

}

int main()

{

    int low, hig;

    cout << "Insert range" << endl << endl;

    cout << "Lowest : " ;

    cin >> low;

    cout << "Highest : " ;

    cin >> hig;

    cout << endl << endl;

    OddPrint result(low,hig);

    result.Print();

    return 0;

}

Question 3

#include <iostream>

#include <string>

using namespace std;

class classified

{

string strName;

int length, num, no = 0, ch = 0, al = 0;

public:

    classified(string, int);

    void process();

    void printNumber(int\*, int);

    void printCharacter(char\*, int);

};

classified::classified(string a, int b)

{

    strName = a;

    length = b;

}

void classified::process()

{

    int numb[length];

    char alpha[length];

    char chara[length];

    for(int nIndex = 0; nIndex < length; nIndex++)

    {

        if(isdigit(strName[nIndex]))

        {

            numb[no] = strName[nIndex] - '0'; //ASCII for 0 = 48

            no++;

        }

        else

        {

            if( (strName[nIndex]>='a' && strName[nIndex]<='z') || (strName[nIndex]>='A' && strName[nIndex]<='Z'))

            {

                alpha[al] = strName[nIndex];

                al++;

            }

            else

            {

                chara[ch] = strName[nIndex];

                ch++;

            }

        }

    }

    cout << "Number = " ;

    printNumber(numb, no);

    cout << "Alphabet = " ;

    printCharacter(alpha, al);

    cout << "Character = " ;

    printCharacter(chara, ch);

}

void classified::printNumber(int arr[], int siz)

{

    for ( int i = 0; i < siz; i++ )

    {

        cout << arr[i];

    }

    cout << endl;

}

void classified::printCharacter(char cha[], int siz)

{

    for ( int i = 0; i < siz; i++ )

    {

        cout << cha[i];

    }

    cout << endl;

}

int main()

{

    string strName;

    cout << "Enter your string : ";

    getline(cin, strName);

    classified cla(strName, strName.length());

    cla.process();

    return 0;

}

Question 4

#include <iostream>

using namespace std;

class convert

{

unsigned int num1;

public:

    convert(unsigned int);

    void Convert();

};

class change

{

unsigned int num, n0, n1, n2, n3;;

public:

    change(unsigned int);

    unsigned int Change();

};

change::change(unsigned int a)

{

    num = a;

}

unsigned int change::Change()

{

    n0 = (num &0x000000FF)>>0;

    n1 = (num &0x0000FF00)>>8;

    n2 = (num &0x00FF0000)>>16;

    n3 = (num &0xFF000000)>>24;

    num = (n0<<24) | (n1<<16) | (n2<<8) | (n3<<0) ;

    cout << endl << "Converted Number : " << dec << num << endl;

    convert cvr(num);

    cvr.Convert();

}

convert::convert(unsigned int a)

{

    num1 = a;

}

void convert::Convert()

{

    cout << "Hexadecimal value : " << hex << num1 << endl;

}

int main()

{

    unsigned int num, \*ptr;

    cout << "Enter the number : ";

    cin >> num;

    ptr = &num;

    convert cvr(\*ptr);

    cvr.Convert();

    change ch(num);

    ch.Change();

}

Question 5

#include <iostream>

using namespace std;

class arrange

{

 int siz;

public:

    string \*day;

    arrange(string\*, int);

    void printString();

};

arrange::arrange(string \*a, int b)

{

    day = new string[b];

    for(int i=0; i<b; ++i)

    {

        day[i] = a[i];

    }

    siz = b;

}

void arrange::printString()

{

    string temp, days[siz], temps;

    for (int i = 0; i < siz; i++)

    {

        for (int k = 0; k < siz; k++ )

        {

            if (day[i] < day[k])

            {

                temp = day[i];

                day[i] = day[k];

                day[k] = temp;

            }

        }

    }

    cout << endl << "After arranged :" << endl;

    for(int nIndex = 0; nIndex < siz; nIndex++)

    {

        cout << day[nIndex] << endl;

    }

}

int main()

{

    string days[7] = {"Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday", "Sunday"};

    cout << "Before arranged :" << endl;

    for(int nIndex = 0; nIndex < 7; nIndex++)

    {

        cout <<days[nIndex] << endl;

    }

    string \*ptr[7];

    for(int nIndex = 0; nIndex < 7; nIndex++)

    {

        ptr[nIndex] = &days[nIndex];

    }

    arrange A(\*ptr, 7);

    A.printString();

    return 0;

}

Question 6

#include <iostream>

#include <vector>

#include <string>

#include <ctype.h>

#include <cstdlib>

using namespace std;

int main()

{

    vector<int> numbers;

    int inp;

    int inp1, tot = 0;

    do{

        cin >> inp;

        numbers.push\_back(inp);

        tot = tot + inp;

    }while(inp!='\r');

    for(int i=0; i < numbers.size()-1; ++i)

        cout << numbers[i] << ' ';

    cout << endl << numbers.size()-1 << endl;

    cout << tot-inp << endl;

    cout << (tot-inp)/(numbers.size()-1) << endl;

    return 0;

}