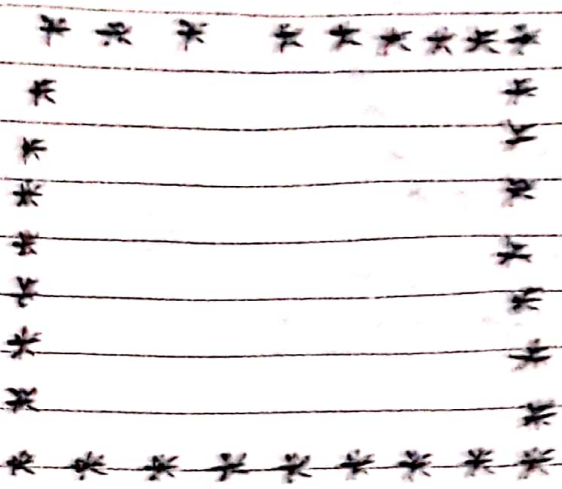
2015 Assignments Tuesday 17

Q1

(a)



```
#include <iostream>
using namespace std;
int main()
{
int i,j;

for(i=1; i<=9; i++)
{
    for(j=1; j<=9; j++)
    {
        if(i==1 || i==9 || j==1 || j==n)
        {
        cout << "*";
        }
    else
        cout << " ";
    }
cout<<"\n";
}
return 0;
}
```

(B)

```
#include <iostream>
using namespace std;
int main()
{
  int i, n, j, k, s=1;
for(i=0; i<=4; i++)
  {
    for (j=n; j>i; j--)
      {
        cout << " ";
      }
  cout << "*";
if (i>0)
      {
        for(k=0; k<=s; k++)
          {
          cout << " ";
          }
        s=s+2;
        cout << "*";
      }
  s=s-4;
  for(i=0; i<=n-1; i++)
  {
    for(j=0; j<=i; j++)
```

My Notes:

```
    {
      cout << " ";
    }
    cout << " *";
  for(k=1; k<=5; k++)
    {
      cout << "*";
    }
  cout << "\n";
  return 0;
}
```

8 Create a class rect. The class has attributes length & width each of which defaults to 1. It has member fn that calculate the perimeter and area of rect. It has set and get fn for both length & breadth. The set fn should verify that length & width are each floating point numbers larger than 0.0 and less than 20.0


```
#include <iostream>
#include <stdio>
using namespace std;
class rectangle.
{
public:
    rectangle (float=1.0, float=1.0)
  float perimeter ( );
  float area ( );
  void set width (float w);
  float get width ( );
  void set length (float l);
  float get length ( );
```

```
private:
        float length;
        float width;
};
Rectangle::Rectangle(float w, float l)
  {
    set width(w);
    set length(l);
  }
float rectangle::perimeter()
  {
    return 2*(length + width);
  }
float rectangle::area()
  {
    return length * width;
  }
void Rectangle::set width(float w)
  {
    width = w>0 && w<20.0 ? w:1.0;
  }
void Rectangle::set length(float l)
  {
    length = l>0 && l<20.0 ? l:1.0;
  }
float Rectangle::get width()
  {
    return width;
  }
float Rectangle::get length()
  {
    return length;
  }
```

My Notes:

```
int main()
{
Rectangle a, b(4.0, 5.0), c(67.0, 888.0);
cout << "length = " << a.getlength()
     << "; width = " << a.getwidth()
     << "; area = " << a.area() <<
     "; perimeter = " << a.perimeter()
                      << endl;
cout << "length = " << b.getlength()
     << "; width = " << b.getwidth()
     << "; area = " << b.area() <<
     "; perimeter = " << b.perimeter() <<
                      endl;

cout << "length = " << c.getlength()
     << "; width = " << c.getwidth()
     << "; area = " << c.area() <<
     "; perimeter = " << c.perimeter <<
                      endl;
return 0;

}
```

Output

length = 1.0 ; width = 1.0 ; perimeter = 4.0 ; area = 1.0

length = 5.0 ; width = 4.0 ; perimeter = 18.0 ; area = 20.0 ;

length = 1.0 ; width = 1.0 ; area = 1.0 ; perimeter = 4.0 ;

Q Create a class called Date that includes three pieces of info as data member - a month (int type), a day (type int) and a year (int type). Your class should have a constructor with three parameters that uses the data parameter to initialize the three data members, assume that values provided for year and date are correct, but ensure that month values in range 1-12, if is not set month to 1. Provide set & get fn for each data member. Provide a member fn display Date that display month, day & year separate forward slashes. Write a test program that demonstrate class date's capabilities

Soln

```
#include <iostream>
#include <string>
using namespace std;

class date.
{
public:
   Date (int, int, int);
   void set month (int);
    int get month ();

   void set day (int);
    int get day ();

   void set year (int);
    int get year ();

   void display Date ();
```

My Notes:

```
private:
    int month;
    int day;
    int year;
};

Date :: Date (int m, int d, int y)
{
    if (m>=1 && m<=12)
        month = m;
    else
    {
        month = 1;
    }
    day = d;
    year = y;
void date :: set month (int m)
{
    if (m>=1 && m<=12)
        month = m;
    else
    {
        month = 1;
    }
}
void date :: set day (int d)
    day = d;
void date :: set year (int y)
    year = y;
```

```
int date :: get month()
{
    return month;
}
int date :: get day()
    return day;
int date :: get year()
    return year;
void date :: display date()
{
cout << month << "/" << day << "/" <<
            year << endl;
}
int main()
{
date d( 12, 9, 2001)
 cout << "Date is :" ;
    d. display date();
    d. set month (0);
    d. set set year (2000);
    d. display date ();
}
```

Q Create an account class that a bank might use to represent customers bank account. Include data member of type int to represent account balance. Provide a constructor that receive an initial balance and uses it to initialize date member. The const should validate initial balance to ensure that it's greater than or equal to zero. If not, set the balance to 0 and display

My Notes:

```
cout << "student-1" << st1.getcount() << endl;
cout << "student-2" << st2.getcount() << endl;
    st1.setcount(6000);

cout << "student-1 after change" << st1.get
        count() << endl;
cout << "student-2 after change" << st2.get
        count() << endl;

}
```

output:

```
count 5000
university IIT
student-1 5000
student-2 5000
student-1 after change 6000
student-2 after " 6000
```

8 create a more sophisticated Rectangle class than one you created in last question. This class stores only cartesian coordinates of four corners of rect. The constructor calls a set fn that accepts four sets of coordinates and verifies that each of these is in first quad with no single nor y coordinate large than 20.0. The set fn also verifies that supplied coordinates do, in fact specify a rect. Number fn calculate length, width, perimeter and area. The length is the longest two dimensions. Include a predicate fn square that determine if the rect is a square.

```
Class Rectangle ;
#include <iostream>
#include <string>

class Rectangle {
public:
    Rectangle (double*, double*, double*,
               double*);
void get coord (double*, double*
                double*, double*);

void perimeter ();
void area ();
void square ();

private:
    double point1 [2];
    double point2 [2];
    double point3 [2];
    double point4 [2];

};
```

My Notes:

```
Rectangle :: Rectangle (double *a, double *b,
                        double *c, double *d)
{
set coord (a,b,c,d);
}

void Rectangle :: set coord (double *p1, double
                             *p2, double *p3,
                             double *p4)
{
const int x=0, y=1;
Point 1[x] = (P1[x] > 20.0 || P1[x] < 0.0)
             ? P1[x] : 0.0;
Point 1[y] = (P1[y] > 20.0 || P1[y] < 0.0)?
             0.0 : P1[y];
Point 2[x] = (P2[x] > 20.0 || P2[x] < 0.0)?
             0.0 : P2[x];
Point 2[y] = (P2[y] > 20.0 || P2[y] < 0.0)?
             0.0 : P2[y];
Point 3[x] = (P3[x] > 20.0 || P3[x] < 0.0)?
             0.0 : P3[x];
Point 3[y] = (P3[y] > 20.0 || P3[y] < 0.0)?
             0.0 : P3[y];
Point 4[x] = (P4[x] > 20.0 || P4[x] < 0.0)?
             0.0 : P4[x];
Point 4[y] = (P4[y] > 20.0 || P4[y] < 0.0)?
             0.0 || P4[y];

if (P1[y] == P2[y] && P1[x] ==
    P4[x] && P2[x] && P3[x] &&
    P3[y] == P4[y])
```

Sunday 15

```
{
    perimeter ();
    area ();
    square ();
}
else
    cout << "coordinate donot form a
            rectangle! \n";
}
void Rectangle :: perimeter(void)
{
    double l = fabs(point4[1] - point1[1]);
    w = fabs(point2[0] - point1[0]);
cout << "length=" << setprecision(1) << (l > w ?
    l : w) << 't' << "width=" (l > w ?
    w : l) << "\n The perimeter is:"
    << 2*(w+l) << '\n'
void Rectangle :: area ()
{
    double l = fabs(point4[1] - point1[1]),
    w = fabs(point2[0] - point1[0]);
cout << "The area is:" << setprecision(1) <<
    w*l << ~~reset~~ "\n\n".
}
void Rectangle :: square()
const int n=0, y=1;
```

My Notes:

```
if (fabs (point1[y] - point1[y])
    == fabs( point2[x] - point1[x]))
cout << "The Rectangle is a square.\n";
}
int main()
{
double w[2] = {1.0, 1.0}, x[2] = {5.0, 1.0},
       y[2] = {5.0, 3.0}, z[2] = {1.0, 3.0};
j[2] = {0.0, 0.0}, k[2] = {1.0, 3.0},
m[2] = {1.0, 1.0}  n[2] = {0.0, 1.0},
v[2] = {99.0, -2.3};
Rectangle a(z, y, x, w), b(j, k, m, n),
          c(w, x, m, n), d(v, n, y, z);
return 0;
}
```