**Competency Assignment Solution**

**Ans. No. 1:**

1. Here apply normalization rule to normalize this register up to 3rd normal form:
2. First Normal Form(1NF):

* Separate multiple values in the service point column.

|  |  |  |  |
| --- | --- | --- | --- |
| Doctor | Contact Number | Service points | Department |
| Dr. Lissa Mwenda | +260766219936  +260766219936  +260766219936 | Antenatal care  Family Planning  Postnatal care | Gynecology |
| Dr. Yvonne Sishuwa | +2607666219936  +2607666219936 | Family Planning  Postnatal care | Pediatrics |
| Dr. Machalo Mbale | +2607745849538 | Antenatal care | Radiology and Imaging |

1. Second Normal Form(2NF):

* Separate table for departments

|  |  |
| --- | --- |
| Doctors | Contact Number |
| Dr. Lissa Mwenda | +2606536434 |
| Dr. Yvonne Sishuwa | +2607263653 |
| Dr. Machalo Mbale | +26037674732 |

* Service points Table:

|  |  |
| --- | --- |
| Doctor | Service Points |
| Dr. Lissa Mwenda | Antenatal care |
| Dr. Lissa Mwenda | Antenatal care |
| Dr. Lissa Mwenda | Postnatal care |
| Dr. Yvonne Sishuwa | Family Planning |
| Dr. Yvonne Sishuwa | Postnatal care |
| Dr. Machalo Mbale | Antenatal care |
| Dr. Machalo Mbale | Radiology and Imaging |

* Department Table:

|  |  |
| --- | --- |
| Doctor | Department |
| Dr. Lissa Mwenda | Gynecology |
| Dr. Yvonne Sishuwa | Pediatrics |
| Dr. Machalo Mbale | Radiology and Imaging |

3 Third Normal Form (3NF)

* Third normal form remove transitive dependencies here I look there is no transitive dependencies in this case, so the tables are in 3NF.

b) The entity relationship diagram (ERD) is looks like this here

Doctor ---< Service Points

|

\---< Department

This notation indicates a “one to many” relationship between doctor and service points, and a “one to many ” relationship between doctor and department.

**Ans. No. 2:**

The value of ‘n’ in each iteration of loop.

Initial Values:

n = 30

i = 0

1. Iteration 1:

n += i (30 + 0) → n = 30

Increment i by 1 (i = 1)

1. Iteration 2:

n += i (30 + 1) → n = 31

Increment i by 1 (i = 2)

1. Iteration 3:

n += i (31 + 2) → n = 33

Increment i by 1 (i = 3)

1. Iteration 4:

n += i (33 + 3) → n = 36

Increment i by 1 (i = 4)

1. Iteration 5:

n += i (36 + 4) → n = 40

Increment i by 1 (i = 5)

1. Iteration 6:

n += i (40 + 5) → n = 45

Increment i by 1 (i = 6)

Loop Exits:

The loop condition i <= 5 is no longer true, so the loop exits.

Print Final Value of n:

The final value of n is 45.

So, the output of the code would be 45.

**Ans. No. 3**

Method Overloading: Multiple methods name with the sane name but different parameter lists is called method overloading.

Here a simple example of method overloading:

Class Addition

{

Public ind Add (int a, int b)

{

return a + b ;

{

Public int Add(int a, int b, int c)

{

return a + b + c;

}

Public double Add(int a, float b)

{

return a + b

}

Public double Add(float a, int b)

{

return a + b:

}

}

Method Overriding: A derived class can be provided a specific implementation for a method that is already defined in its base class.

Here is the simple example of method overriding

class Parent

{

public virtual void Display()

{

Console.WriteLine("Parent's Display method");

}

}

class Child : Parent

{

public override void Display()

{

Console.WriteLine("Child's Display method");

}

}

class Program

{

static void Main()

{

Parent parent = new Parent();

parent.Display();

Child child = new Child();

child.Display();

Parent anotherChild = new Child();

anotherChild.Display();

}

}