Unit 2

Q) Explain function Prototype with example.

Ans. Function Prototype (ફંક્શન પ્રોટોટાઈપ) એ ફંક્શનની આર્ગ્યુમેન્ટ ની રાઈપ, જેટલી આર્ગ્યુમેન્ટ હોય તે અને તેની ડેટા રાઈપ ની માહિતી કમ્પાઈલરને આપે છે.

- આ એક બીજું નામ છે જે ફંક્શન ડિક્લેરેશન અને ડેફિનિશન બંનેને યુઝ થાય છે.
- જ્યારે ફંક્શન કોલ થાય ત્યારે કમ્પાઈલર પ્રોટોટાઈપનો ઉપયોગ કરીને ખાતરી કરે છે કે ફંક્શનની આર્ગ્યુમેન્ટ અને રિટર્ન રાઈપ બરાબર છે કે નહિ.
- ફંક્શન પ્રોટોટાઈપ એ ડિક્લેરેશન ઓફ ફંક્શન છે.
- Syntax:

return_type Function_name (argument list);

- argument list આ આર્ગ્યુમેન્ટની રાઈપ અને નંબર ફંક્શનનો પાસ કરે છે.

Example:

```
int sum (int x, int y);
```

- ફંક્શન પ્રોટોટાઈપમાં આર્ગ્યુમેન્ટના ડેટા ટાઈપ અને વેરીએબલ નામ - નામ આપવામાં આવે છે.
- ફંક્શન ડેફિનેશનમાં આર્ગ્યુમેન્ટના નામ ફરજીયાત આપવા પડે છે

↳ Example:

```
int sum (int, int);
```

- ફંક્શન કોલિંગમાં, આર્ગ્યુમેન્ટના નામ ફરજીયાત નથી.

↳ Example:

```
int sum (int a, int b)
{
    int total = a+b;
    return (total);
}
```

Q2

Call by Reference	Call by Value
- Copy (સમાનાર્થી) રેફરન્સ નો ઉપયોગ actual argument માટે કરવામાં આવે છે	- અહિં ~~a~~ actual argument નો જ ઉપયોગ કર્યા રેફરન્સ તરીકે થાય છે.
- extra મેમરિનો ઉપયોગ કરવાની જરૂર નથી પડતી.	- રેફરન્સની કોપી સ્ટોર કરવા માટે extra મેમરીની જરૂર પડે છે.
- કોલ કરેલું ફંક્શન actual value પર કરી શકે છે.	- કોલ કરેલું ફંક્શન actual ~~fi~~ value નો ઉપયોગ નથી કરતુ.
- ~~સ્પીડ~~ વધારે છે.	- સ્પીડ ઓછી છે.
- કોમ્પલ્સ અને ઈઝી છે અને ~~C++~~ C++ પ્રોગ્રામમાં વપરાય છે.	- કોમ્પલ્સ અને ઈઝી છે. અને ~~C~~ C પ્રોગ્રામમાં અને C++ પ્રોગ્રામમાં વપરાય છે.
- One way communication પ્રોવાઈડ કરે છે.	- Two way communication પ્રોવાઈડ કરે છે.
-	

Q3 Inline function

- પ્રોગ્રામમાં એક જ ફંક્શનની કોડ એક કરતાં વધારે વખત ઉપયોગ કરી શકાય છે

- જેના લીધે પ્રોગ્રામની સાઈઝ નાની થઈ જાય છે અને સ્પીડ વધે છે.

- પરંતુ જો આ પ્રક્રિયામાં ફંક્શન કોપી અને વિકન વારંવાર કરવામાં આવે તો ઓવરહેડ વધી જાય છે અને બફરમાં સમય બગાડે છે.

- C++ પ્રોગ્રામની અંદર Inline function ની સુવિધા છે કે ફંક્શન અને મેક્રો બંને તરીકે કામ કરે છે.

Syntax:

```
inline return_type function_name
                (arguments list)
{
    // body of the function
}
```

- Inline શબ્દ નો ઉપયોગ ફંક્શનની ડેફિનિશન માં મોખરે પહેલો કરવામાં આવે છે.

- જ્યારે (Inline) શબ્દ કંપાઈલરને

request મોકલે છે જેવાથી કંપાઇલરની body માં કંપાઇલરની કેટલીકરણની જરૂરથી execute (એક્ઝેક્યુટ) થાય છે.

example:

```
# include <iostream.h>
const float pie = 3.14;
inline float area (float r)
{
    return (pie * r * r);
}

void main ()
{
    float radius;

    cout << "enter radius";
    cin >> radius;
    cout << " Area = ";
    cout << area (radius);
```

Q4 Static Member Function.

- Static. Keyword નો ઉપયોગ કરીને member function ને static બનાવવામાં આવે છે.

Characteristics:

- ~~કોઈ પણ~~ કોઈપણ ક્લાસ કોને મેમ્બરનો કે બીજા ક્લાસ મેમ્બર ફંક્શનનો જ ઉપયોગ કરી શકે છે.

- તે ક્લાસના નામનો અને કોલન ઓપરેટરનો ઉપયોગ કરીને કોલ કરી શકાય છે.

Syntax:

```
class_name :: function_name;
```

Example:

```
class test
{ private:
    int code;
    static int count;

  public:
    void setcode();
    void showcode();
    static void showcount()
    {
      cout << " count = " << count;
    }
};
```

```
void main()
{
    ...
    ...
    test :: showcount();
}
```

Q 5 Define a class Employee; having

data members : (1) Emp_no (2) Name
member function : (1) getdata() (2) putdata()

```
class Employee
{
  private:
      int Emp_no;
      char Emp_name[20];
  Public:
      void getdata()
      {
          cout << " enter employee
                  information \n";
          cin >> Emp_no;
          cin >> Emp_name;
      }
      void putdata()
      {
          cout << " Employee no = "
```

```
    cout << Emp_no;
    cout << " Employee Name"
    cout << Emp_name;
  }
};
```

Q-6 Function Overloading

C++ પ્રોગ્રામમાં એક કરતા વધારે ફંક્શનનું નામ સરખું રાખીને અલગ અલગ code રન કરી શકાય છે. આને ફંક્શન ઓવરલોડીંગ કહે છે.

- કંપાઈલર ફંક્શનના નંબર ઓફ આર્ગ્યુમેન્ટ અને ટાઈપ પરથી ફંક્શનને અલગ પાડે છે.
- ફંક્શન સીગ્નેચર નીચે પ્રમાણે કરવામાં આવે છે.

```
float area ( float r);
float area ( float l, float b);
```

- Circle નો Area શોધવા માટે પહેલુ area ફંક્શન તૈયાર કરો

- જ્યારે Rectangle નો Area શોધવા માટે બીજું ફંક્શન તૈયાર કરો.

- ફંક્શન તૈયાર કરવા માટે,

↳ a = area (radius);

↳ a = area (length, breadth);

- પહેલુ ફંક્શન Circle માટે Area શોધશે.
- બીજું ફંક્શન Rectangle માટે Area શોધશે.

9-7 Default Argument.

- C++ પ્રોગ્રામમાં ફંક્શનને ડિફોલ્ટ કર્યા પછી આ ફંક્શનમાં આર્ગ્યુમેન્ટ કોલ કરી શકાય છે.

- તેના માટે ડિફોલ્ટ આર્ગ્યુમેન્ટનો ઉપયોગ કરવામાં આવે છે.

- જો ડિફોલ્ટ કર્યા કરતાં આ ફંક્શનમાં આર્ગ્યુમેન્ટ વધુ ફંક્શનને કોલ કરવામાં આવે તો કંપાઈલર

Default આર્ગ્યુમેન્ટનો ઉપયોગ કરીને ફંક્શન બનાવી શકે છે.

- Default આર્ગ્યુમેન્ટ બનાવવા માટે ફંક્શન ડિક્લેર કરતી વખતે જ આર્ગ્યુમેન્ટની ડિફોલ્ટ વેલ્યુ પાસ કરવામાં આવે છે.

eg. void set_point (int x, int y=0);

- ઉપરના ફંક્શનમાં y ની ડિફોલ્ટ વેલ્યુ 0 છે.

- set_point (4);

- ઉપરના ફંક્શન કોલ સ્ટેટમેન્ટમાં એક જ વેલ્યુ પાસ કરી છે. તો બીજા વેરિએબલની ડિફોલ્ટ વેલ્યુનો કેવા ઈલિટ ઉપયોગ કરશે. જેની x = 4 અને y = 0 થશે.

- set_point (4, 6);

- ઉપરના સ્ટેટમેન્ટમાં બંને વેલ્યુ પાસ કરી છે જેની x = 4 અને y = 6 થશે.

Q-8 Constant Argument.

- const keyword નો ઉપયોગ કરીને પ્રોગ્રામમાં ફંક્શનની આર્ગ્યુમેન્ટ ને constant બનાવવામાં આવે છે.
- જે પેરામીટરને constant બનાવવામાં આવ્યો હોય તેની વેલ્યુ આખા પ્રોગ્રામ દરમિયાન બદલાતી નથી.

eg

```
int fun (const int x)
{
    int y;
    x = x+1; // invalid
    y = x; // valid
    return (y);
}
```

- ઉપરના example પ્રમાણે x ની વેલ્યુ change નથી કરી શકાતી પરંતુ તેનો ઉપયોગ કરી શકાય છે.
- Advantage: constant argument ની વેલ્યુનું આકસ્મિક રીતે ફેરફાર થતી રોકવું અને તે Secure રહે છે.

Q9 Friend Function.

- ક્લાસની બહારના ફંક્શનને ક્લાસનો Friend બનાવવા માટે Friend Function નો ઉપયોગ થાય છે.
- એના માટે ફંક્શનની ડેફિનેશનમાં Friend keyword (શબ્દ) લખવો જરૂરી છે.
- જે ફંક્શન Friend તરીકે ડિક્લેર થયેલું હોય તે આખા પ્રોગ્રામમાં સાદા ફંક્શન તરીકે પણ ઉપયોગી છે.

* Characteristics :

- તે જે ક્લાસમાં Friend તરીકે ડિક્લેર થયેલું છે તેના scope (સ્કોપ) માં નથી હોતું.
- તેને ક્લાસના ઓબ્જેક્ટના ઉપયોગ કર્યા વગર કોલ કરી શકાય છે.
- તે ક્લાસના મેમ્બરને, તે ક્લાસના ઓબ્જેક્ટના નામ અને ડોટ ઓપરેટર થી જ કોલ કરી શકે છે.
- તેને પ્રાઇવેટ અથવા પબ્લીકમાં ડિક્લેર કરી શકાય છે.
- તેમાં ઓબ્જેક્ટ આર્ગ્યુમેન્ટ તરીકે પસાર થાય છે.

Example:

```
class ABC;
class XYZ
{ private:
    int a;
  public:
    void setvalue(int x)
    { a=x;
    }
    friend void max(ABC, XYZ);
};
class ABC
{ private:
    int b;
  Public:
    void setvalue(int y)
    { b=y;
    }
    friend void max(ABC, XYZ);
};
void max(ABC p, XYZ q)
{
    if(p.b > q.a)
    {
      cout << "b is maximum";
    }
    else
    {
      cout << "a is max";
    }
}
```