Welcome To

Basic Application Development in Python

Subject Code : 28531

3rd Semester,

Computer Technology

ফাংশন (Function)

কোন প্রোগ্রাম যখন অনেক বড় তখন সেই প্রোগ্রাম লেখা এবং এর ব্যবস্থাপনা করা অনেক কঠিন হয়ে পড়ে।

আবার কখনো কখনো প্রোগ্রাম একই কাজ বারবার করতে হয় এবং এজন্য একই স্টেট বারবার লিখতে হয়।

এসব সমস্যা সমাধানের জন্য একটি গুরুত্বপূর্ন মাধ্যম হলো ফাংশন।

ফাংশন (Define a Function)

ফাংশন : পাইথনে ফাংশন হলো অর্গানাইজ ও রিইউজ্যাবল কোডের একটি নির্দিষ্ট ব্লক যা একটি প্রোগ্রাম একটি স্বাধীন ক্ষুদ্র অংশ , যার একটি নাম থাকে , যা এক বা একাধিক স্টেমেন্টের সমন্বয়ে গঠিত এবং নির্দিষ্ট কোন সমস্যা সমাধানের জন্য ব্যবহৃত হয়।

ফাংশন ব্যবহারের সুবিধাসমূহ (Advantages of Function):ফাংশন ব্যবহার ফলে অনেকগুলো সুবিধা পাওয়া যায়। যেমন :

- ফাংশন ব্যবহার এর ফলে প্রোগ্রামের কোনো অংশ মূল প্রোগ্রাম বার বার লিখতে প্রয়োজন হয় না।
- প্রোগ্রাম সাইজ ছোট হয় ।
- প্রোগ্রাম লিখতে অপেক্ষাকৃত কম সময় লাগে।
- প্রোগ্রাম ভুল নির্ণয় ও সংশোধন করা সহজ হয়।

বড় প্রোগ্রাম অনেকগুলো ফাংশন মডিউল ভাগ করে নিয়ে এক এক মডিউল আলাদাভাবে বিভিন্ন প্রোগ্রামের দিয়ে প্রোগ্রাম করা হলে এবং এগুলো ফাংশন হিসেবে ব্যবহার করা হলে স্বল্পতম সময়ে প্রোগ্রাম সম্পন্ন করা যায়। অত্যন্ত জরুরি প্রোগ্রামকে ফাংশন হিসেবে লাইব্রেরি জমা করে রাখলে অন্য ব্যবহারকারী প্রয়োজনে তা ব্যবহার করতে পারে।

1.	**abs()**	**Returns absolute value of a number**
2.	all()	Returns true when all element in iterable is true
3.	any()	Checks if any Element of an Iterable is True
4.	bin()	Converts a integer to binary string
5.	Dir()	Tries to Return Attributes of object
6.	Divmod()	Returns an Enumerate Object

7.	**float()**	**Returns floating point number from number ,string**
8.	help()	Invokes the built-in Help System
9.	id()	Returns Identify of an Object
10.	len()	Returns Length of an Object
11.	mas()	Returns langest element
12.	min()	Returns smallest element
13.	next()	Retrieves Next Element from Iterator
14.	Open()	Returns a File object
15.	Print()	Prints the Griven Object
16.	Set()	Returns a set

17.	**Str()**	**Returns informal representation of an object**
18.	Sum()	Add items of an Iterable
19.	Super()	Allow you to Refer Parenr Class by super
20.	Pow()	Returns x to the power of y
21.	Property()	Returns a property attribute
22.	Setattr()	Sets value of an attribute of object
23.	Slicer()	Creates a slice object specifited by ranger()
24.	Sorted()	Returns sorted list from a given iterable

25.	Tuple()Funtion	Creates a Tuple
26.	Type()	Returns type of an Object
27.	Zip()	Returns an Iterator of Tuples
28.	Vars()	Returns_dict_attribute of a class
29.	List() Funtion	Creates list in
30.	_import_()	Advanced Funtion Called by import

ইউজার ডিফাইন্ড ফাংশন (user defined function): ইউজার বা ব্যবহারকারী তার নিজস্ব প্রয়োজন এবং প্রঙ্গা অনুযায়ী যেসব ফাংশন ডিফাইন করে প্রোগ্রামে ব্যবহার করেন সে সকল ফাংশনকে ইউজার ডিফাইন্ড ফাংশন বলে। ইউজার ডিফাইন্ড ফাংশন আকার আকৃতি ও সমস্যার ধরণও সমাধানের কৌশলের উপর নির্ভর করে।একটি নির্দিষ্ট কাজের জন্য ভিন্ন ভিন্ন পোগ্রামার ভিন্ন ভিন্ন নামে ইউজার ডিফাইন করতে পারেন।ইউজার ডিফাইন্ড ফাংশনের নামকরণের বেলায় পোগ্রামারকে বেশ সচেতন হতে হয়।

ইউজার ডিফাইন্ড ফাংশন ব্যবহার সুবিধা (Benefits of User Defined Function):

পাইথনের ইউজার ডিফাইন্ড ফাংশন ব্যবহারের নানাবিধ সুবিধা রয়েছে। যেমনঃ

উপযুক্ত স্থানে ফাংশনের ব্যবহার করতে পারলে প্রোগ্রামের আকারে ছোট হয়ে যায়।

প্রোগ্রামের ভুল-ত্রুটি নিণয় ও সংশোধন সহজতর হয়।

প্রোগ্রম এক্সিকিউশনে সময় কম লাগে।

একটি প্রোগ্রাম ব্যবহৃত ফাংশনকে সহজেই অন্য প্রোগ্রামেও ব্যবহার করা যায়। ফলে উক্ত ফাংশন বডিকে বারবার রিপিট করার প্রয়োজন করার প্রয়োজন হয়না।

রিকাসন পদ্ধতিতে একটি ফাংশন প্রয়োজনে নিজে নিজ নিজেই পুনঃ পুনঃ এক্সেস করতে পারে।

ইউজার ডিফাইন্ড ফাংশন ব্যবহার করে একটি বড় প্রোগ্রামকে ছোট ছোট স্বাধীন প্রোগ্রামে রুপান্তর করা যায় বলে প্রোগ্রামটি অধিক সহজতর ও আকষনীয় বলে প্রতিয়মান হয়।

ফাংশনের প্রকারভেদ types of function

অন্যন্য প্রোগ্রামিং ল্যাংগুয়েজ মত পাইথনে ফাংশনে ও দুই ভাগে ভাগ করা যায়

বিল্ট ইন ফাংশন বা লাইব্রেরি ফাংশন

ইউজার ডিফাইন্ড ফাংশন

বিল্ট ইন ফাংশন বা লাইব্রেরি ফাংশনঃ যে সকল ফাংশন ডেভেলপার কতৃর্ক পূব থেকেই নিধারিত কাজের জন্য ডিফাইনকৃত , কম্পাইলারের আওতাভুক্ত এবং নির্দিষ্ট নাম সম্পূর্ন ,ব্যবহারকারী চাইলে যাদের নাম পরিবর্তন করতে পারেন না সে সকল ফাংশনকে লাইব্রেরী ফাংশন বলে।

আরগুমেন্ট ভেরিয়েবল :

কোন ইউজার ডিফাইন্ড ফাংশনের প্রথম বন্ধনি বা প্যারেনথিসিসের মধ্যে কোন ভেরিয়েবল ঘোষনা করা হলে ঐ ভেরিয়েবল বা ভেরিয়েবলসমূহকে প্যারামিটার বলে। কোন ফাংশনের এক বা একাধিক প্যারামিটার ব্যবহার করা

অ্যানোনিমাস ফাংশন : ল্যাম্বডা :

পাইথনে lambda অপারেটর ব্যবহার করে এক লাইনের যে ফাংশন তৈরি করা হয় তাকে ল্যাম্বডা ফাংশন বলে।

একে অ্যানোনিমাস ফাংশন ও বলে। সাধারনত : def কীওয়াড ব্য স্বয়ত্রিয় ভাবে এটিকে একটি ভেরিয়েবলে অ্যাসাইন করে দেওয়া হয় যার মাধ্যমে একে পরবর্তীতে কল করা যায়।

ল্যাম্বাডা ফাংশন ঘোষনা :

ল্যাম্বাডা ফাংশন ঘোষনার ক্ষেত্রে lambda এর পর স্পেস দিয়ে আর্গুমেন্ট দিতে হয়। তারপর কোলন চিহ্ন দিয়ে ল্যাম্বডা ফাংশনের ডিফাইন করতে হয়। যেমন ঃ lambda x,y:x+y

উদাহরন

```
sum=lambda x,y:x+y
print(sum(30,20))
print(lambda x,y:x+y)(30,20)
```

আউটপুট

```
50
50
```

যেতে পারে। তবে প্রতিটি আরগুমেন্ট ভেরিয়েবল মাঝে কমা দিতে হবে। আর যখন একটি ফাংশনের কল করা হয় তখন সেই ফাংশনের প্যারামিটার হিসেবে যে ভ্যালু পাঠানো হয় তাকে আর্গুমেন্ট বলা হয়।

ফাংশনের ব্যবহৃত প্যারামিটার আরগুমেন্ট প্রকারভেদ :

পাইথন ফাংশনের ব্যবহৃত প্যারামিটার চার প্রকার।

- রিকোয়ার্ড আরগুমেন্ট
- কীওয়াড আরগুমেন্ট
- ডিফল্ট আরগুমেন্ট
- ভেরিয়েবল আরগুমেন্ট

কীওয়াড আরগুমেন্ট : যে সকল আরগুমেন্ট কোন ফাংশনের সঠিক পজিশনাল অর্ডারে পাস করা হয় তাদেরকে কীওয়াড কীওয়ার্ড আরগুমেন্ট বলে।এক্ষেত্রে ফাংশন ডেফিনিশেনের আরগুমেন্ট সংখ্যা এবং কল্ড ফাংরনের আরগুমেন্ট সংখ্যা সমান হবে।

উদাহরন :

```
def add(a,b,c) :
return (a+b+c)
temp=add(1,2)
print (temp)
```

আউটপুট ঃ

```
Traceback ( most recent call lost ):
File "/home/ugcoder/Desktop/test.py", line 3 in <module>
temp=add(1,2)
TypeError:add()missing 1 required positional agument:'c'
```

তারিখ ও সময় ফাংশনের ব্যবহার (Uses of Date Time Functions) :

পাইথন প্রোগ্রামে তারিখ ও সময়কে () নিয়ে বিভিন্ন দরণের অপারেশন সম্পাদনের জন্য পাচটি মডিউল রয়েছে। যথা ঃ

date

time

datetime

timedelta এবং

tzinfo

নিম্নে এ সকল মডিউলের বিবরণ বণনা করা হলোঃ

মডিউল	বিবরণ
date	শুদুমাএ তারিখ (date) কে ম্যানিপুলেট (Month,day,year) করার জন্য এই মডিউল ব্যবহার করা হয় ।
time	সময় (Hour,minute,second,microsecond) কে ম্যানিপুলেট করার জন্য এই মডিউল ব্যবহার করা হয় ।
datetime	তারিখ ও সময় উভয়কে একএ (Month, day, year, hour, second, microsecond) ম্যানিপুলেট করার জন্য এই মডিউল ব্যবহার করা হয়।
timedelta	নিদিস্ট ডিউরেশনের সময়কে ম্যানিপুলেট করার জন্য এই ব্যবহার করা হয় ।
tzinfo	নিদিস্ট টাইম জোন নিয়ে কাজ করার জন্য এই মডিউল ব্যবহার করা হয় ।

ভেরিয়েবলের স্কোপ (Scope of Variables) :

ভেরিয়েবলের Scope এর উপর ভি িও করে ভেরিয়েবল প্রদানত দুই প্রকার।

যথা :

লোকাল ভেরিয়েবল (Local veriable) ও

গ্লোবাল ভেরিয়েবল (Global variable)

লোকাল ভেরিয়েবল (Local variable) : যেসব ভেরিয়েবল কাযকারিতা শুদুমাএ কোনো নিদিষ্ট অংশ বা ফাংশনের মদ্যে সীমাবন্দ থাকে তাকে ভেরিয়েবল বলে।

গ্লোবাল ভেরিয়েবল (Global variable): যখন কোনো ভেরিয়েবলের কাযকারিতা কোনো নিদিষ্ট ফাংশনের মদ্যে সীমাবব্যদ না থেকে এর মান সকল ফাংশনেই ব্যবহার করা যায় তাকে গ্লোবাল ভেরিয়েবল বলে।

রিকাসিভ ফাংশন (Recursive Function) :

প্রোগ্রাম কোনো ফাংশন নিজেই নিজেকে Call করতে পারে। কোনো ফাংশন নিজেই নিজেকে যখন Call করে তখন সেই ফাংশঙ্কে রিকাসিভ ফাংশন (Recuresive Function) বলে। আর ফাংশন কতক নিজেই নিজেকে কল করার এই প্রএয়াটিকে রিকাসন (Recursion) বলে। রিকাসিব ফাংশন (Recursive function) এর এমন কোনো শত দেয়া থাকে, যাতে প্রদত্ত শতের ভিত্তিতে রিকাসন প্রএয়া চলতে থাকে। তা না হলে ফাংশনের রিপিটিশন চলতেই থাকবেই।

উদাহরণ।ঃ

```
def factorial(n):
    if n == 1:
        return 1
    else:
        return n * factorial(n-1)
```

```
print (factorial(3))
def Area():
length=int(input("Enter the value of Length:"))
width=int(input("Enter the value of Width:"))
RectangleArea=length*width
prinr("Area of Rectangle is=",RectangleArea)
Area()
 def Largest():
```

```
Number1=int(input("Enter the value of 1st Number:"))
Number2=int(input("Enter the value of 2nd Number :"))
if(Number1>Number2):
print("Largest is Number1 & it is=",Number1)
else:
print("Largest is Number2 & it is=",Number2)
Largest()
def multiply(numbers):
total=1
for x in number :
total *= x
return total
print(multiply((8,2,3,10,7)))
```

The End