**COMPARATIVE STUDY(POWERSHELL/SHELL/PYTHON)**

|  |  |  |  |
| --- | --- | --- | --- |
| Command Name | PowerShell | Shell | Python |
| Output Statement | Write-host  eg: Write-host “hello”  for printing value-  write-host” hello$value” | echo  eg: echo (“hello”)  for printing value:  echo -e (“hello$value”) | print  eg: print(“hello”)  for printing value -print (“{}”. format(value)) |
| Input Statement | Read-Host  eg: Read-Host “enter Your name”. for taking variable input-  $value = Read-Host "enter your name"  For taking no. input  [int] $number = Read-Host "enter no" | Read -p  eg: read -p “enter your name”. for taking variable input-  read -p "Enter Number:" num1 | Input  eg: input (“enter name”)  for taking variable input-  num=input(”enter a no.”) |
| Arithmetic operators | **+,-,\*,/**  **Eg: a + b** | **+,-,\*,/**  **Eg: a + b** | **+,-,\*,/**  **Eg: a + b** |
| Logical Operators | And ,or, xor, not  **Eg: -and ,-or** | And ,Or  **&& , ||** | And, or  **Eg: and , or** |
| Comparison Operator | -eq (equals to)  -ne (not equals to)  -gt (greater)  -ge(greater equal to)  -lt(lesser)  -le(lesser equal to)  -like  -notlike  -match  -notmatch  -in | = / -eq  !=  > / -gt  >= /-ge  < /lt  <=/-le | ==  !=  >  >=  <  <= |
| Conditional Statements | If (condition)  {  Statements  }  elseif  {  Statements  }  Switch ($value)  1  { statements  }  2  {  }  3{}  Default{}  Try-Catch-Finally:  Try  { line of code }  Catch[exceptions]  { set of actions }  Finally  {  whatever is the result, then finally statement will execute  } | If [ condition ]  then  Statements  elif [ condition ]  then  statements  fi  case ($value) in  1)    Statements  ;;  \*)  Statements  ;;    esac | if(conditions):  Statements  elif(conditions):  statements  else: |
| Looping Statements | **While:**  while(conditions)  { statements }  eg: while($i -le 20)  { …… }  **For:** for(initialization;condition;repeat)  { statement }  eg: for($i=1;$i -le 20;i++)  { ……. }  **Switch:** switch (value)  1 { … }  2 { ……}  Default  {…. }  **Foreach:**  foreach($item in $collection)  { statements } | **While:**  while [ conditions ]  Do  Statements  Done  eg: while [ $i -le $range ]  do  ……….  Done  **For:**  for (( i=1; i<=$end; i++ ))  do  ……..  done | **While:**  while(conditions): …statements…….  **For:**  for i in range(1,100)  …statements….. |
| Functions: | Function Name  {  … Function definition…  }  To call: Name | Name()  {  …Function definition…  }  To call: Name | def Name():  …function definition…..  To call: Name() |
| Coding Example | cls  Write-Host " multiplication table using while" -BackgroundColor Cyan  $i=1  while ($i -le 20)  {  Write-Host "10 x $i" = (10\*$i)  $i++  } | clear  echo "Multiplication Table"  read -p "enter the starting point" i  read -p "enter the end point" j  read -p "enter the no which you want to multiply with " num  while [ $i -le $j ]  do  result=$((num\*i))  echo -e "$num \* $i = $result"  ((i=i+1))  done | print("Number Guessing game")  import random  randomv=(random.randint(0,99))  ##print(randomv)  for i in range(1,5):  num=input("Enter the no. you guessed :")  diff=abs(int(randomv)-int(num))  if(diff>5 and diff<10):  print("you are closer")  #print(diff)  elif(diff>10):  print("you are far away")  elif(diff>0 and diff<5):  print("you are very closer")  elif(diff==0):  print("You are correct" )  break  print("correct no .is {}".format(randomv)) |