Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 1

1) Write a program to demonstrate inheritance

CODE:

```
#include "iostream"
using namespace std;

class Parent{
public:
    void displayParent(){
        cout << "From Daddy" << endl;
    }
};

class Child : public Parent{
public:
    void displayChild(){
        cout << "From Boomer" << endl;
    }
};

int main(){
    Child child;
    child.displayParent();
    child.displayChild();
    return 0;
}
```

OUTPUT:

```
[Running] cd "d:\Degree\SEM 3\PCPF\Programs\"
From Daddy
From Boomer

[Done] exited with code=0 in 0.833 seconds
```

2) Write a program to demonstrate Encapsulation

CODE:

```
#include "iostream"
#include "string"
using namespace std;

class Character{
private:
    string name;
public:
    void getName(string n){
        name = n;
    }
    void printName(){
        cout << "Name: " << name << endl;
    }
};
int main(){
    Character p;
    p.getName("Arthur Morgan");
    p.printName();
    return 0;
}
```

OUTPUT:

```
[Running] cd "d:\Degree\SEM 3\PCPF\Programs\"
Name: Arthur Morgan

[Done] exited with code=0 in 0.953 seconds
```

Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 2

1) Write a program to demonstrate constructor and destructor

CODE:

```
#include "iostream"
#include "string"
using namespace std;

class Character
{
private:

    string name;

public:

    Character(string n)
    {
        cout << "Character is in scope - Object created" << endl;
        name = n;
        cout << name << " is a a damn good character" << endl;
    }

    ~Character()
    {
        cout << name << " is probably dead" << endl;
        cout << "Character is now out of scope - Object destroyed" << endl;
    }
```

```
};

int main()
{
    Character T("Trevor Philips");
    Character J("John Marston");
    return 0;
}
```

OUTPUT:

```
[Running] cd "d:\Degree\SEM 3\PCPF\Programs\" && g++ exp2.c++ -o exp2
Character is in scope - Object created
Trevor Philips is a a damn good character
Character is in scope - Object created
John Marston is a a damn good character
John Marston is probably dead
Character is now out of scope - Object destroyed
Trevor Philips is probably dead
Character is now out of scope - Object destroyed

[Done] exited with code=0 in 0.89 seconds
```

Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 3

1) Implement a program in SWI Prolog to find factorial of a number.

CODE:

```
factorial(0, 1).  % Base case: factorial of 0 is 1
factorial(N, Result) :-
    N > 0,         % Ensure N is greater than 0
    X is N - 1,    % Decrement N
    factorial(X, Y),  % Recursive call to calculate factorial of N-1
    Result is N * Y.  % Result is N multiplied by factorial of N-1

% Specify what should be executed at startup
:- initialization(main).

main :-
    factorial(5, Result),   % Calculate the factorial of 5
    write('Factorial is: '),
    write(Result).
```

OUTPUT:

```
Output:

Factorial is: 120
```

Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 4

1) Implement a program in SWI Prolog to solve Tower of Hanoi puzzle.

CODE:

```
% Base case: Move 1 disk from X to Y
move(1, X, Y, _) :-
    write('Move top disk from '),
    write(X), write(' to '), write(Y), nl.

% Recursive case: Move N disks
move(N, X, Y, Z) :-
    N > 1,
    M is N - 1,
    move(M, X, Z, Y),    % Move M disks from X to Z using Y as auxiliary
    move(1, X, Y, _),    % Move the remaining disk from X to Y
    move(M, Z, Y, X).    % Move M disks from Z to Y using X as auxiliary

% Specify what should be executed at startup
:- initialization(main).

% Main predicate to run Tower of Hanoi
main :-
    write('Solving Tower of Hanoi for 3 disks'), nl,
    move(3, 'Left', 'Right', 'Center'),  % Solve for 3 disks
    halt.
```

OUTPUT:

Output:

```
Solving Tower of Hanoi for 3 disks
Move top disk from Left to Right
Move top disk from Left to Center
Move top disk from Right to Center
Move top disk from Left to Right
Move top disk from Center to Left
Move top disk from Center to Right
Move top disk from Left to Right
```

Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 5

1) Write a program in SWI-Prolog environment to print Fibonacci series.

CODE:

```
% Base case: Fibonacci of 0 is 0, and Fibonacci of 1 is 1
fibonacci(0, 0).
fibonacci(1, 1).

% Recursive case: Fibonacci(N) = Fibonacci(N-1) + Fibonacci(N-2)
fibonacci(N, Result) :-
    N > 1,
    N1 is N - 1,
    N2 is N - 2,
    fibonacci(N1, Res1),
    fibonacci(N2, Res2),
    Result is Res1 + Res2.

% Specify what should be executed at startup
:- initialization(main).

% Main predicate to run Fibonacci calculation
main :-
    write('Fibonacci of 10 is: '), nl,
    fibonacci(10, Result),   % Calculate Fibonacci of 10
    write(Result), nl,
    halt.
```

OUTPUT:

Output:

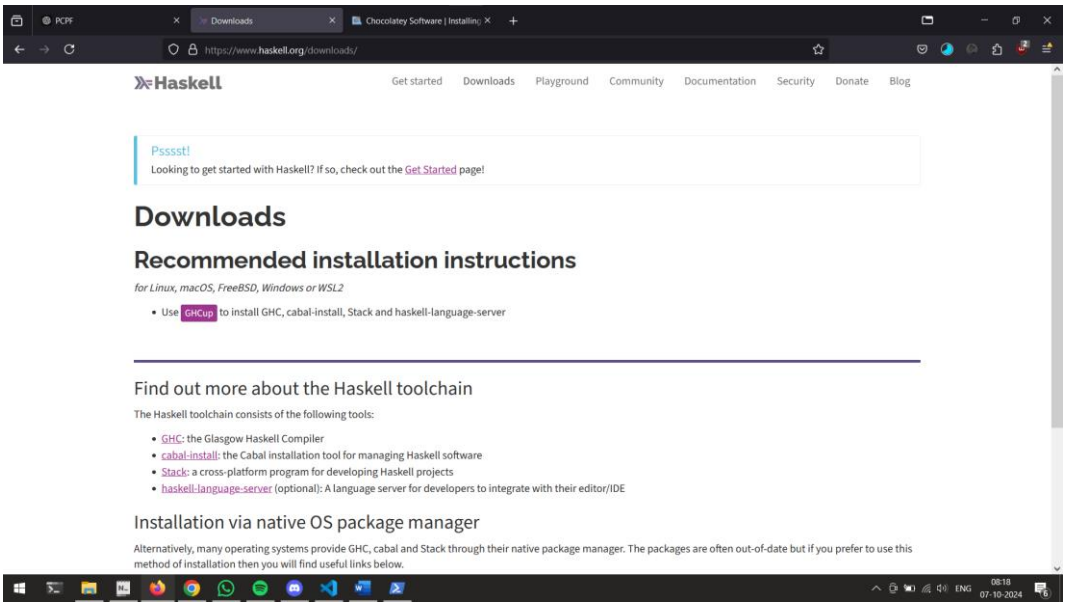
```
Fibonacci of 10 is:
55
```

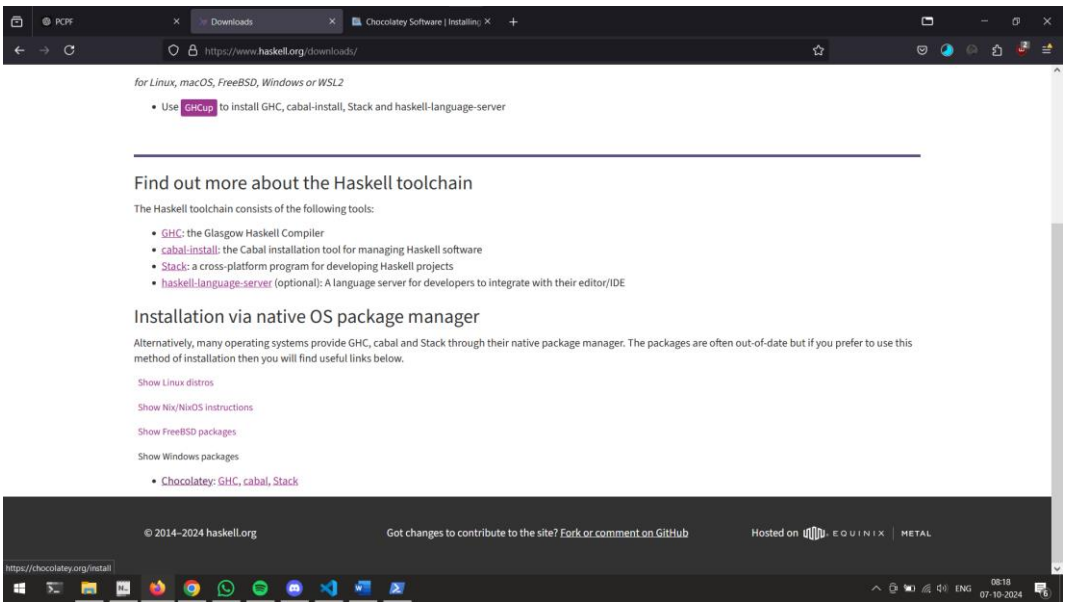
Name: Abdurrahman Qureshi

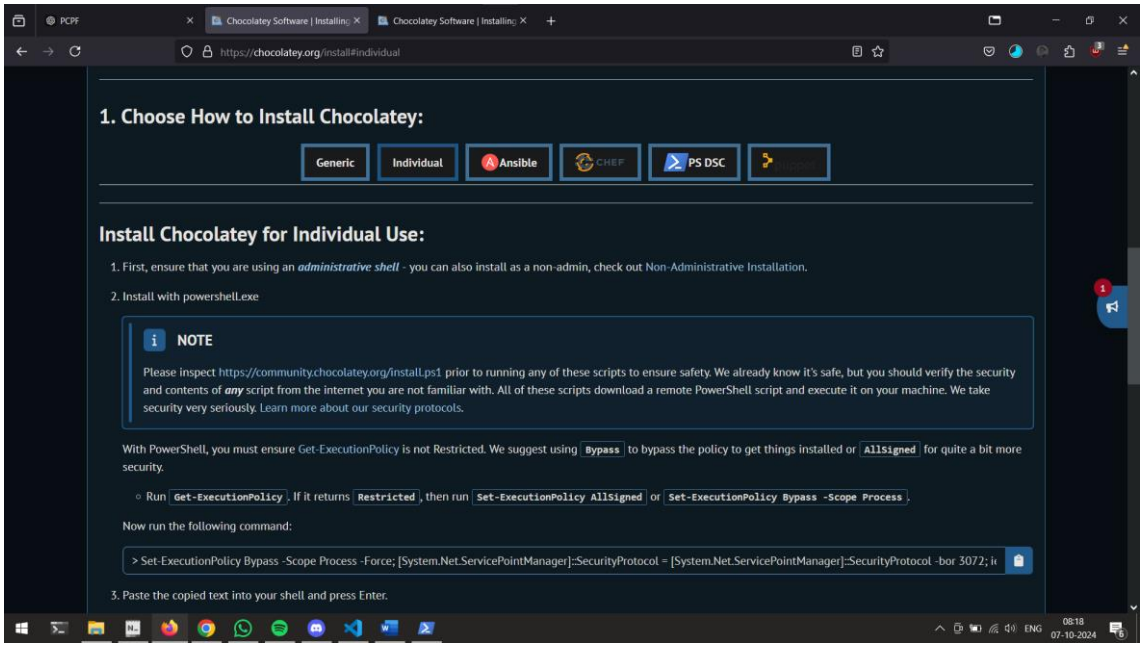
Roll No: 242466

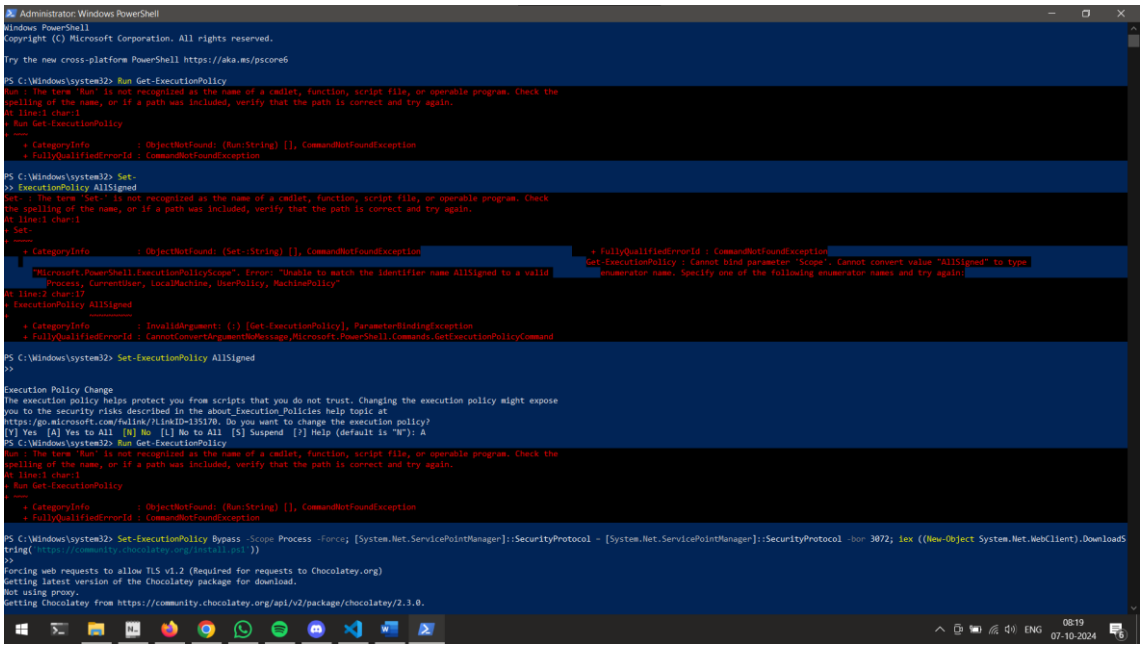
Practical No: 6

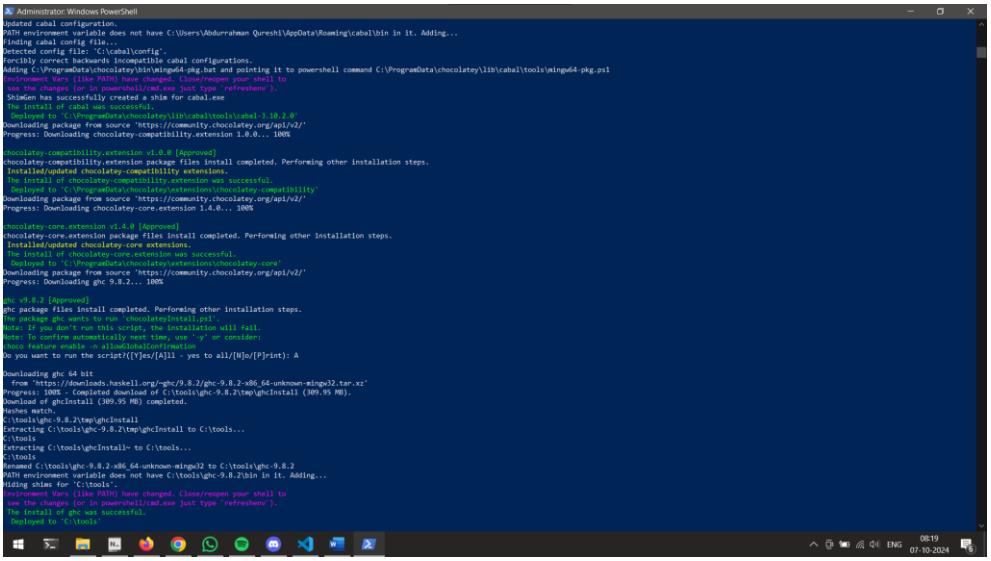
1) Write down steps for installing Haskell compiler on Windows 10.

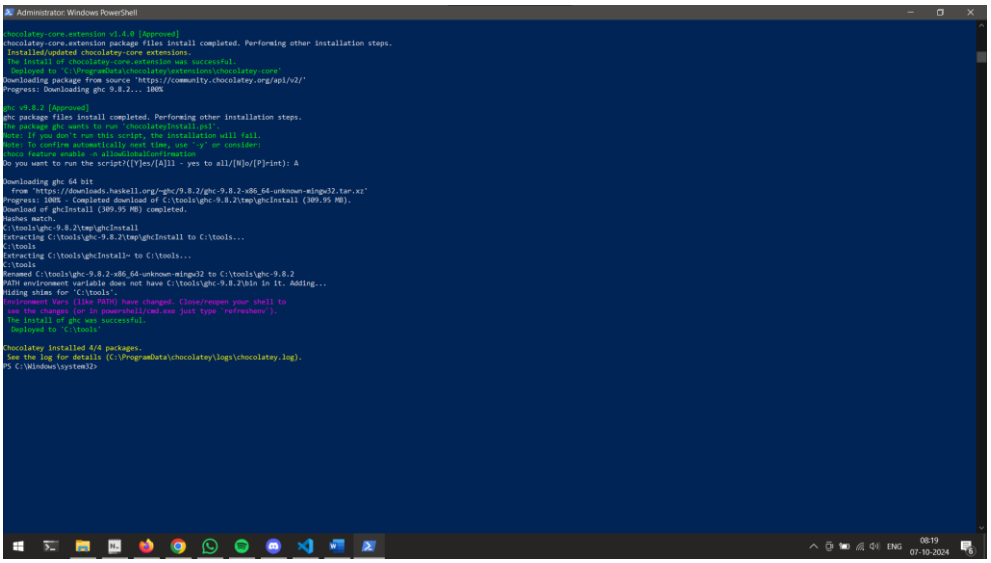


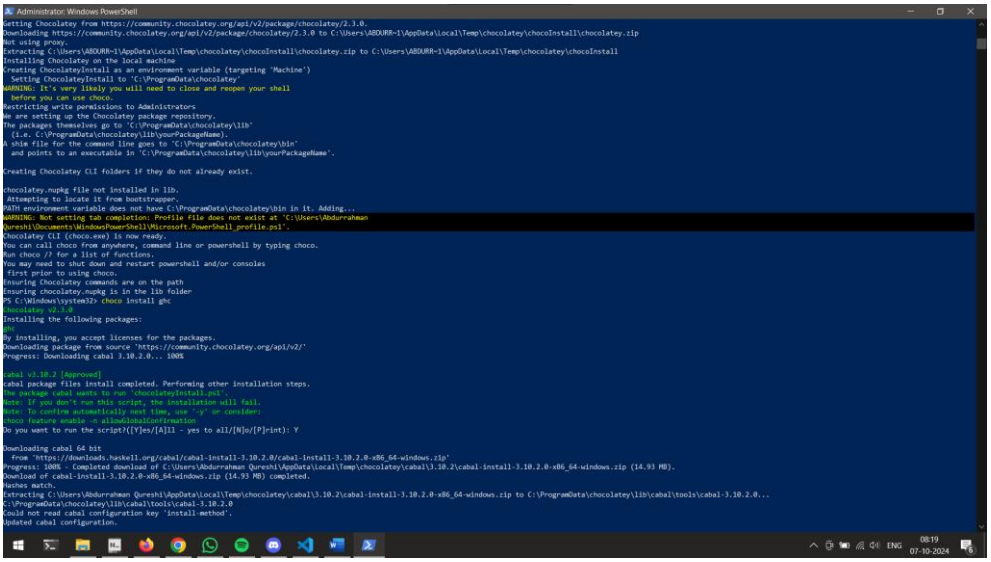



PCPF
Chocolatey Software | Installing
Chocolatey Software | Installing
https://chocolatey.org/install#individual
1. Choose How to Install Chocolatey:
Generic
Individual
Ansible
CHEF
PS DSC
Install Chocolatey for Individual Use:
1. First, ensure that you are using an administrative shell - you can also install as a non-admin, check out Non-Administrative Installation.
2. Install with powershell.exe
NOTE
Please inspect https://community.chocolatey.org/install.ps1 prior to running any of these scripts to ensure safety. We already know it's safe, but you should verify the security and contents of any script from the internet you are not familiar with. All of these scripts download a remote PowerShell script and execute it on your machine. We take security very seriously. Learn more about our security protocols.
With PowerShell, you must ensure Get-ExecutionPolicy is not Restricted. We suggest using Bypass to bypass the policy to get things installed or AllSigned for quite a bit more security.
Run Get-ExecutionPolicy. If it returns Restricted, then run Set-ExecutionPolicy AllSigned or Set-ExecutionPolicy Bypass -Scope Process.
Now run the following command:
> Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; ie
3. Paste the copied text into your shell and press Enter.
ENG
08:18
07-10-2024


Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Windows\system32> Run Get-ExecutionPolicy
Run : The term 'Run' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify that the path is correct and try again.
At line:1 char:1
+ Run Get-ExecutionPolicy
+ ~~~
+ CategoryInfo : ObjectNotFound: (Run:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
PS C:\Windows\system32> Set-
>> ExecutionPolicy AllSigned
Set- : The term 'Set-' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify that the path is correct and try again.
At line:1 char:1
+ Set-
+ ~~~~
+ CategoryInfo : ObjectNotFound: (Set-:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
Get-ExecutionPolicy : Cannot bind parameter 'Scope'. Cannot convert value "AllSigned" to type "Microsoft.PowerShell.ExecutionPolicyScope". Error: "Unable to match the identifier name AllSigned to a valid enumerator name. Specify one of the following enumerator names and try again:
Process, CurrentUser, LocalMachine, UserPolicy, MachinePolicy"
At line:2 char:17
+ ExecutionPolicy AllSigned
+ ~~~~~~~~~
+ CategoryInfo : InvalidArgument: (:) [Get-ExecutionPolicy], ParameterBindingException
+ FullyQualifiedErrorId : CannotConvertArgumentNoMessage,Microsoft.PowerShell.Commands.GetExecutionPolicyCommand
PS C:\Windows\system32> Set-ExecutionPolicy AllSigned
>>
Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose you to the security risks described in the about_Execution_Policies help topic at
https://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): A
PS C:\Windows\system32> Run Get-ExecutionPolicy
Run : The term 'Run' is not recognized as the name of a cmdlet, function, script file, or operable program. Check the spelling of the name, or if a path was included, verify that the path is correct and try again.
At line:1 char:1
+ Run Get-ExecutionPolicy
+ ~~~
+ CategoryInfo : ObjectNotFound: (Run:String) [], CommandNotFoundException
+ FullyQualifiedErrorId : CommandNotFoundException
PS C:\Windows\system32> Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))
>>
Forcing web requests to allow TLS v1.2 (Required for requests to Chocolatey.org)
Getting latest version of the Chocolatey package for download.
Not using proxy.
Getting Chocolatey from https://community.chocolatey.org/api/v2/package/chocolatey/2.3.0.
ENG
08:19
07-10-2024


Administrator: Windows PowerShell
Updated cabal configuration.
PATH environment variable does not have C:\Users\Abdurrahman Qureshi\AppData\Roaming\cabal\bin in it. Adding...
Finding cabal config file...
Detected config file: 'C:\cabal\config'.
Forcibly correct backwards incompatible cabal configurations.
Adding C:\ProgramData\chocolatey\bin\mingw64-pkg.bat and pointing it to powershell command C:\ProgramData\chocolatey\lib\cabal\tools\mingw64-pkg.ps1
Environment Vars (like PATH) have changed. Close/reopen your shell to
see the changes (or in powershell/cmd.exe just type `refreshenv`).
ShimGen has successfully created a shim for cabal.exe
The install of cabal was successful.
Deployed to 'C:\ProgramData\chocolatey\lib\cabal\tools\cabal-3.10.2.0'
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading chocolatey-compatibility.extension 1.0.0... 100%
chocolatey-compatibility.extension v1.0.0 [Approved]
chocolatey-compatibility.extension package files install completed. Performing other installation steps.
Installed/updated chocolatey-compatibility extensions.
The install of chocolatey-compatibility.extension was successful.
Deployed to 'C:\ProgramData\chocolatey\extensions\chocolatey-compatibility'
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading chocolatey-core.extension 1.4.0... 100%
chocolatey-core.extension v1.4.0 [Approved]
chocolatey-core.extension package files install completed. Performing other installation steps.
Installed/updated chocolatey-core extensions.
The install of chocolatey-core.extension was successful.
Deployed to 'C:\ProgramData\chocolatey\extensions\chocolatey-core'
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading ghc 9.8.2... 100%
ghc v9.8.2 [Approved]
ghc package files install completed. Performing other installation steps.
The package ghc wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): A
Downloading ghc 64 bit
from 'https://downloads.haskell.org/~ghc/9.8.2/ghc-9.8.2-x86_64-unknown-mingw32.tar.xz'
Progress: 100% - Completed download of C:\tools\ghc-9.8.2\tmp\ghcInstall (309.95 MB).
Download of ghcInstall (309.95 MB) completed.
Hashes match.
C:\tools\ghc-9.8.2\tmp\ghcInstall
Extracting C:\tools\ghc-9.8.2\tmp\ghcInstall to C:\tools...
C:\tools
Extracting C:\tools\ghcInstall= to C:\tools...
C:\tools
Renamed C:\tools\ghc-9.8.2-x86_64-unknown-mingw32 to C:\tools\ghc-9.8.2
PATH environment variable does not have C:\tools\ghc-9.8.2\bin in it. Adding...
Hiding shims for 'C:\tools'.
Environment Vars (like PATH) have changed. Close/reopen your shell to
see the changes (or in powershell/cmd.exe just type `refreshenv`).
The install of ghc was successful.
Deployed to 'C:\tools'
ENG
08:19
07-10-2024


Administrator: Windows PowerShell
chocolatey-core.extension v1.4.0 [Approved]
chocolatey-core.extension package files install completed. Performing other installation steps.
Installed/updated chocolatey-core extensions.
The install of chocolatey-core.extension was successful.
Deployed to 'C:\ProgramData\chocolatey\extensions\chocolatey-core'
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading ghc 9.8.2... 100%
ghc v9.8.2 [Approved]
ghc package files install completed. Performing other installation steps.
The package ghc wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): A
Downloading ghc 64 bit
from 'https://downloads.haskell.org/~ghc/9.8.2/ghc-9.8.2-x86_64-unknown-mingw32.tar.xz'
Progress: 100% - Completed download of C:\tools\ghc-9.8.2\tmp\ghcInstall (309.95 MB).
Download of ghcInstall (309.95 MB) completed.
Hashes match.
C:\tools\ghc-9.8.2\tmp\ghcInstall
Extracting C:\tools\ghc-9.8.2\tmp\ghcInstall to C:\tools...
C:\tools
Extracting C:\tools\ghcInstall= to C:\tools...
C:\tools
Renamed C:\tools\ghc-9.8.2-x86_64-unknown-mingw32 to C:\tools\ghc-9.8.2
PATH environment variable does not have C:\tools\ghc-9.8.2\bin in it. Adding...
Hiding shims for 'C:\tools'.
Environment Vars (like PATH) have changed. Close/reopen your shell to
see the changes (or in powershell/cmd.exe just type `refreshenv`).
The install of ghc was successful.
Deployed to 'C:\tools'
Chocolatey Installed 4/4 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
PS C:\Windows\system32>
ENG
08:19
07-10-2024


Administrator: Windows PowerShell
Getting Chocolatey from https://community.chocolatey.org/api/v2/package/chocolatey/2.3.0.
Downloading https://community.chocolatey.org/api/v2/package/chocolatey/2.3.0 to C:\Users\ABDURR~1\AppData\Local\Temp\chocolatey\chocoInstall\chocolatey.zip
Not using proxy.
Extracting C:\Users\ABDURR~1\AppData\Local\Temp\chocolatey\chocoInstall\chocolatey.zip to C:\Users\ABDURR~1\AppData\Local\Temp\chocolatey\chocoInstall
Installing Chocolatey on the local machine
Creating ChocolateyInstall as an environment variable (targeting 'Machine')
Setting ChocolateyInstall to 'C:\ProgramData\chocolatey'
WARNING: It's very likely you will need to close and reopen your shell
before you can use choco.
Restricting write permissions to Administrators
We are setting up the Chocolatey package repository.
The packages themselves go to 'C:\ProgramData\chocolatey\lib'
(i.e. C:\ProgramData\chocolatey\lib\yourPackageName).
A shim file for the command line goes to 'C:\ProgramData\chocolatey\bin'
and points to an executable in 'C:\ProgramData\chocolatey\lib\yourPackageName'.
Creating Chocolatey CLI folders if they do not already exist.
chocolatey.nupkg file not installed in lib.
Attempting to locate it from bootstrapper.
PATH environment variable does not have C:\ProgramData\chocolatey\bin in it. Adding...
WARNING: Not setting tab completion: Profile file does not exist at 'C:\Users\Abdurrahman
Qureshi\Documents\WindowsPowerShell\Microsoft.PowerShell_profile.ps1'.
Chocolatey CLI (choco.exe) is now ready.
You can call choco from anywhere, command line or powershell by typing choco.
Run choco /? for a list of functions.
You may need to shut down and restart powershell and/or consoles
first prior to using choco.
Ensuring Chocolatey commands are on the path
Ensuring chocolatey.nupkg is in the lib folder
PS C:\Windows\system32> choco install ghc
Chocolatey v2.3.0
Installing the following packages:
ghc
By installing, you accept licenses for the packages.
Downloading package from source 'https://community.chocolatey.org/api/v2/'
Progress: Downloading cabal 3.10.2.0... 100%
cabal v3.10.2 [Approved]
cabal package files install completed. Performing other installation steps.
The package cabal wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[A]ll - yes to all/[N]o/[P]rint): Y
Downloading cabal 64 bit
from 'https://downloads.haskell.org/cabal/cabal-install-3.10.2.0/cabal-install-3.10.2.0-x86_64-windows.zip'
Progress: 100% - Completed download of C:\Users\Abdurrahman Qureshi\AppData\Local\Temp\chocolatey\cabal\3.10.2\cabal-install-3.10.2.0-x86_64-windows.zip (14.93 MB).
Download of cabal-install-3.10.2.0-x86_64-windows.zip (14.93 MB) completed.
Hashes match.
Extracting C:\Users\Abdurrahman Qureshi\AppData\Local\Temp\chocolatey\cabal\3.10.2\cabal-install-3.10.2.0-x86_64-windows.zip to C:\ProgramData\chocolatey\lib\cabal\tools\cabal-3.10.2.0...
C:\ProgramData\chocolatey\lib\cabal\tools\cabal-3.10.2.0
Could not read cabal configuration key 'install-method'.
Updated cabal configuration.
ENG
08:19
07-10-2024

BASIC ARITHMETIC OPERATIONS

```
MINGW64:/c/Windows/Syster

Abdurrahman Qureshi@GhouledGadget MINGW64 /c/Windows/System32
$ gchi
bash: gchi: command not found

Abdurrahman Qureshi@GhouledGadget MINGW64 /c/Windows/System32
$ ghci
GHCi, version 9.8.2: https://www.haskell.org/ghc/  :? for help
ghci> 2 + 5
7
ghci> 2-5
-3
ghci> 2*7
14
ghci> 3/9
0.3333333333333333
ghci> 3/3
1.0
ghci> 9/3
3.0
ghci> 85*2
170
ghci> 100-90
10
ghci> 50+25
75
ghci>
```

LOGICAL OPERATIONS

```
MINGW64:/c/Windows/Syster

Abdurrahman Qureshi@GhouledGadget MINGW64 /c/Windows/System32
$ ghci
GHCi, version 9.8.2: https://www.haskell.org/ghc/  :? for help
ghci> false

<interactive>:1:1: error: [GHC-88464]
    Variable not in scope: false
    Suggested fix:
      Perhaps use data constructor 'False' (imported from Prelude)
ghci> False
False
ghci> True
True
ghci> False && True
False
ghci> True && True
True
ghci> False && False
False
ghci> False || True
True
ghci> True || True
True
ghci> False || False
False
ghci> not (False && True)
True
ghci> not (False || True)
False
```

COMPARATIVE OPERATIONS

```
MINGW64:/c/Windows/Syster

Abdurrahman Qureshi@GhouledGadget MINGW64 /c/Windows/System32
$ ghci
GHCi, version 9.8.2: https://www.haskell.org/ghc/  :? for help
ghci> 69==96
False
ghci> 56==56
True
ghci> 5<9
True
ghci> 0>8
False
ghci> 0==0
True
ghci> not (100<52)
True
ghci> not (100>52)
False
ghci>
```

Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 7

1) Demonstrate the use of inbuilt functions in Haskell.

OUTPUT:

```
MINGW64:/c/Windows/Syster

Abdurrahman Qureshi@GhouledGadget MINGW64 /c/Windows/System32
$ ghci
GHCi, version 9.8.2: https://www.haskell.org/ghc/  :? for help
ghci> succ (69)
70
ghci> succ (succ (69))
71
ghci> max 40 30
40
ghci> max 40 300
300
ghci> succ 56 + max 40 300
357
ghci> div 90 5
18
ghci> div 2 5
0
ghci> 4/0
Infinity
ghci> mod 8 3
2
ghci> mod 100 4
0
ghci>
```

1) Create a Haskell script and print "Hello world"

CODE:

```
main :: IO()
main = do
    putStrLn "Hello, world!"
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MSYS /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7.exe
Hello, world!
```

2) Basic arithmetic operations on integer

CODE:

```
main :: IO()
main = do
  putStrLn "Hello, world!"
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
Addition of 7 + 9:
16
Subtraction of 9 - 7:
2
Product of 7 * 9:
63
Division of 9 / 7:
1.2857143
```

2) Basic arithmetic operations on floats

CODE:

```
main :: IO()
main = do
  putStrLn "Hello, world!"
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
Addition of 7 + 9:
16.0
Subtraction of 9 - 7:
2.0
Product of 7 * 9:
63.0
Division of 9 / 7:
1.2857143
```

3) Print a user defined Fibonacci series

CODE:

```
main :: IO ()
main = do
  putStrLn "Enter the number of terms:"
  n <- readLn
  print (take n (fibonacci 0 1))

fibonacci :: Int -> Int -> [Int]
fibonacci a b = a : fibonacci b (a + b)
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
Enter the number of terms:
6
[0,1,1,2,3,5]

Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
Enter the number of terms:
12
[0,1,1,2,3,5,8,13,21,34,55,89]
```

4) Use the map function on a list [1..5] to produce a list [2..6]

CODE:

```
main :: IO ()
main = print (map (+1) [1..5])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
[2,3,4,5,6]
```

5) Use the map function on a list [1..5] to produce a list [2..6]

CODE:

```
main :: IO ()

main = print (map (*2) [1..10])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
[2,4,6,8,10,12,14,16,18,20]
```

6) Use the map function on a list [1..5] to produce a list [2..6]

CODE:

```
main :: IO ()

main = print (filter odd [1..30])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
[2,4,6,8,10,12,14,16,18,20]
```

7) Use the map function on a list [1..5] to produce a list [2..6]

CODE:

```
main :: IO ()

main = print (filter (< 4) [1..10])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
[1,2,3]
```

8) Haskell program To double a number

CODE:

```
main :: IO ()
```

```
main = print (filter (< 4) [1..10])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
10
```

9) Haskell program To perform multiplication of 2 numbers

CODE:

```
main :: IO ()

main = print (filter (< 4) [1..10])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
24
```

10) Haskell program To find the numbers in range 1 to 20 that are divisible by 3

CODE:

```
main :: IO ()

main = print (filter (< 4) [1..10])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp7
[3,6,9,12,15,18]
```

Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 8

1) Demonstrate the use of basic list operations.

CODE:

```
main :: IO()
main = do
  putStrLn "Increment each element by 1:"
  print (map (+1) [1, 2, 3, 4, 5])

  putStrLn "Filter elements greater than 3:"
  print (filter (>3) [1, 2, 3, 4, 5])

  putStrLn "Filter even numbers from the list:"
  print (filter even [1, 2, 3, 4, 5])

  putStrLn "Filter odd numbers from the list:"
  print (filter odd [1, 2, 3, 4, 5])

  putStrLn "Get the length of the list:"
  print (length [1, 2, 3, 4, 5])

  putStrLn "Get the head of the list:"
  print (head [1, 2, 3, 4, 5])

  putStrLn "Get the tail of the list:"
  print (tail [1, 2, 3, 4, 5])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp8.exe
Increment each element by 1:
[2,3,4,5,6]
Filter elements greater than 3:
[4,5]
Filter even numbers from the list:
[2,4]
Filter odd numbers from the list:
[1,3,5]
Get the length of the list:
5
Get the head of the list:
1
Get the tail of the list:
[2,3,4,5]
```

2) Demonstrate a program for Pattern Matching

CODE:

```
main :: IO()
main = do
   let numbers = [1, 2, 3, 4, 5]
   print (sumList numbers)

sumList :: [Int] -> Int
sumList [] = 0
sumList (x:xs) = x + sumList xs
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp8.exe
15
```

2) Demonstrate a program for List Comprehension

CODE:

```
main :: IO()
main = do
  putStrLn "Double elements greater than 2:"
  print ([x * 2 | x <- [1..5], x > 2])
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp8.exe
Double elements greater than 2:
[6,8,10]
```

Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 9

1) Exception handling

C++ CODE:

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

int main(){
    int divisor, dividend;
    cout << "Enter dividend: ";
    cin >> dividend;
    cout << "Enter divisor: ";
    cin >> divisor;
    try{
        if (divisor == 0)  {
            throw runtime_error("Division by zero error");
        }
        cout << "Result: " << dividend / divisor << endl;
    }
    catch (runtime_error &e){
        cout << "Exception caught: " << e.what() << endl;
    }
    cout << "Press Enter to continue...";
    cin.ignore();
    cin.get();

    return 0;
}
```

OUTPUT:

```
MINGW64:/d/Degree/SEM 3/F

Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp9.exe
Enter dividend: 8
Enter divisor: 5
Result: 1

0

Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ ./exp9.exe
Enter dividend: 9
Enter divisor: 0
Exception caught: Division by zero error

Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$
```

JAVA CODE:

```
public class exp9 {
    public static double divide(int numerator, int denominator) throws IllegalArgumentException {
        if (denominator == 0) {
            throw new IllegalArgumentException("Division by zero is not allowed.");
        }
        return (double) numerator / denominator;
    }
    public static void main(String[] args) {
        int num = 10;
        int denom = 0;

        try {
            double result = divide(num, denom);
            System.out.println("Result: " + result);
        } catch (IllegalArgumentException e) {
            System.err.println("Error: " + e.getMessage());
        }
    }
}
```

OUTPUT:

```
Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$ java exp9
Error: Division by zero is not allowed.
```

2) Garbage Collection.

C++ CODE:

```
#include <iostream>
```

```
class MyClass {
public:
    MyClass() { std::cout << "Constructor called!" << std::endl; }
    ~MyClass() { std::cout << "Destructor called!" << std::endl; }
};
int main() {
        MyClass obj = MyClass();
    std::cout << "End of the program." << std::endl;
    return 0;
}
```

OUTPUT:

```
[Running] cd "d:\Degree\SEM 3\PCPF\Programs\" && g++ exp9.c++ -o exp9 && "d:\Degree\SEM 3\PCPF\Programs\"exp9
Constructor called!
Destructor called!
End of the program.
```

JAVA CODE:

```
public class exp9 {
    static class MyClass {
        MyClass() {
            System.out.println("Constructor called!");
        }
    }
    public static void main(String[] args) {
        MyClass obj = new MyClass();
        System.out.println("Auto destructor called.");
    }
}
```

OUTPUT:

```
[Running] cd "d:\Degree\SEM 3\PCPF\Programs\" && javac exp9.java && java exp9
Constructor called!
Auto destructor called.
```

Name: Abdurrahman Qureshi

Roll No: 242466

Practical No: 10

1) Write a tic-tac-toe program in 3 different languages (Programming Assignment For comparative study of Different Paradigms)

JAVA CODE:

```
import java.util.Scanner;

public class m {
    private char[][] board;
    private char currentPlayer;

     m() {
        board = new char[3][3];
        currentPlayer = 'X';
        initializeBoard();
    }

    public void initializeBoard() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                board[i][j] = '-';
            }
        }
    }

    public void printBoard() {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
```

```
            System.out.print(board[i][j] + " ");
        }
        System.out.println();
    }
}

public boolean isBoardFull() {
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            if (board[i][j] == '-') {
                return false;
            }
        }
    }
    return true;
}

public boolean checkForWin() {
    for (int i = 0; i < 3; i++) {
        if (board[i][0] == currentPlayer && board[i][1] == currentPlayer && board[i][2] == currentPlayer) {
            return true;
        }
        if (board[0][i] == currentPlayer && board[1][i] == currentPlayer && board[2][i] == currentPlayer) {
            return true;
        }
    }
    if (board[0][0] == currentPlayer && board[1][1] == currentPlayer && board[2][2] == currentPlayer) {
        return true;
    }
    if (board[0][2] == currentPlayer && board[1][1] == currentPlayer && board[2][0] == currentPlayer) {
        return true;
    }
    return false;
}

public void changePlayer() {
    currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
}

public void play() {
    Scanner scanner = new Scanner(System.in);
```

```
        while (true) {
            System.out.println("Current board:");
            printBoard();
            System.out.println("Player " + currentPlayer + ", enter your move (row and column): ");
            int row = scanner.nextInt();
            int col = scanner.nextInt();

            if (row < 0 || row >= 3 || col < 0 || col >= 3 || board[row][col] != '-') {
                System.out.println("This move is not valid");
                continue;
            }

            board[row][col] = currentPlayer;

            if (checkForWin()) {
                System.out.println("Player " + currentPlayer + " wins!");
                printBoard();
                break;
            }

            if (isBoardFull()) {
                System.out.println("The game is a draw!");
                printBoard();
                break;
            }

            changePlayer();
        }
        scanner.close();
    }

    public static void main(String[] args) {
        m game = new m();
        game.play();
    }
}
```

OUTPUT:

```
MINGW64:/d/Degree/SEM 3/F
1
This move is not valid
Current board:
- - -
- - -
- O X
Player X, enter your move (row and column):
1
1
Current board:
- - -
- X -
- O X
Player O, enter your move (row and column):
1
2
Current board:
- - -
- X O
- O X
Player X, enter your move (row and column):
0
0
Player X wins!
X - -
- X O
- O X

Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
$
```

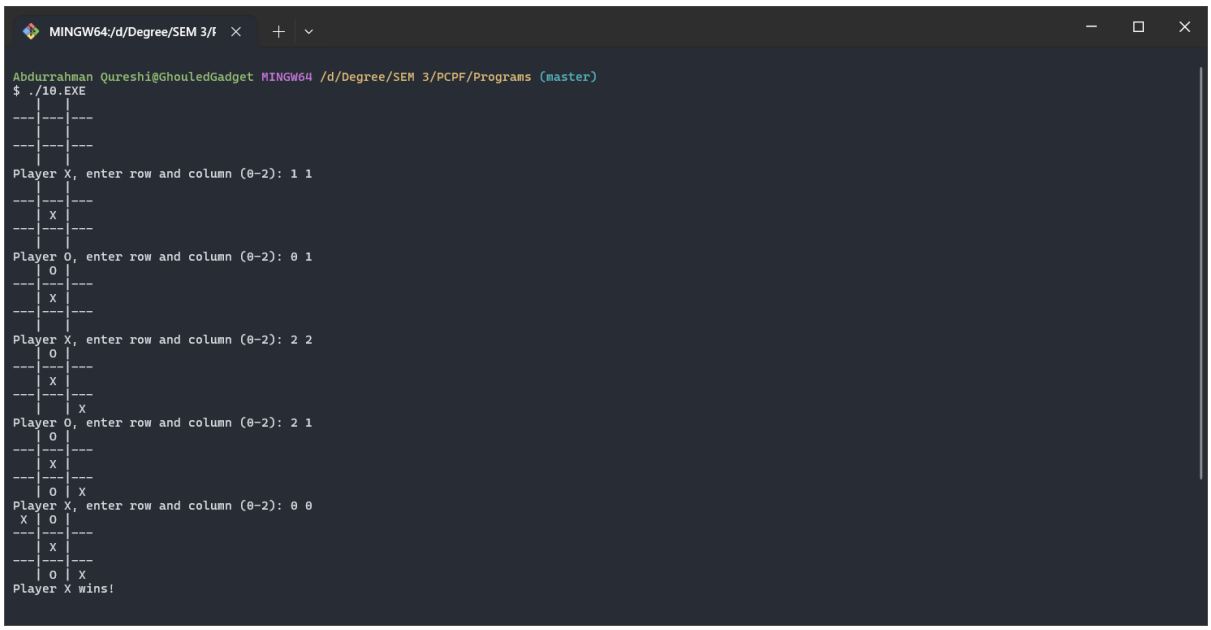
HASKELL CODE:

```
import Data.List (intersperse, transpose)
import Data.Char (digitToInt)
import Text.Read (readMaybe)

type Board = [[Char]]

-- Initialize an empty board
emptyBoard :: Board
emptyBoard = replicate 3 (replicate 3 ' ')

-- Display the board
printBoard :: Board -> IO ()
printBoard b = putStrLn $ unlines $ map (intersperse '|') b

-- Check if a player has won
checkWin :: Board -> Char -> Bool
checkWin board player =
  let rows = board
      cols = transpose board
      diag1 = [board !! i !! i | i <- [0..2]]
      diag2 = [board !! i !! (2 - i) | i <- [0..2]]
  in any (all (== player)) (rows ++ cols ++ [diag1, diag2])
```

```
-- Check if the board is full
isFull :: Board -> Bool
isFull = all (all (/= ' '))

-- Make a move on the board
makeMove :: Board -> Int -> Int -> Char -> Board
makeMove board row col player =
  take row board ++ [take col (board !! row) ++ [player] ++ drop (col + 1) (board !! row)] ++ drop (row + 1)
board

-- Function to parse user input
parseInput :: String -> Maybe (Int, Int)
parseInput input =
  case words input of
    [r, c] -> do
      row <- readMaybe r
      col <- readMaybe c
      if row >= 0 && row <= 2 && col >= 0 && col <= 2
        then return (row, col)
        else Nothing
    _ -> Nothing

-- Main function to play the game
main :: IO ()
main = playGame emptyBoard 'X'

-- Recursive function to continue the game
playGame :: Board -> Char -> IO ()
playGame board player = do
  printBoard board
  if checkWin board (if player == 'X' then 'O' else 'X')
    then putStrLn $ "Player " ++ [if player == 'X' then 'O' else 'X'] ++ " wins!"
    else if isFull board
      then putStrLn "It's a draw!"
      else do
        putStrLn $ "Player " ++ [player] ++ ", enter row and column (0-2) separated by a space (e.g., '1 2'):"
        input <- getLine
        case parseInput input of
          Just (row, col) ->
            if board !! row !! col /= ' '
```

```
            then putStrLn "Invalid move, try again." >> playGame board player
            else playGame (makeMove board row col player) (if player == 'X' then 'O' else 'X')
        Nothing -> putStrLn "Invalid input, try again." >> playGame board player
```

OUTPUT:



C++ CODE:

```
#include <stdio.h>

char board[3][3];
char currentPlayer = 'X';

void initBoard() {
    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 3; j++)
            board[i][j] = ' ';
}

void printBoard() {
    printf(" %c | %c | %c\n", board[0][0], board[0][1], board[0][2]);
    printf("---|---|---\n");
```

```
    printf(" %c | %c | %c\n", board[1][0], board[1][1], board[1][2]);
    printf("---|---|---\n");
    printf(" %c | %c | %c\n", board[2][0], board[2][1], board[2][2]);
}

int checkWin() {
    for (int i = 0; i < 3; i++) {
        if (board[i][0] == currentPlayer && board[i][1] == currentPlayer && board[i][2] == currentPlayer) return 1;
        if (board[0][i] == currentPlayer && board[1][i] == currentPlayer && board[2][i] == currentPlayer) return 1;
    }
    if (board[0][0] == currentPlayer && board[1][1] == currentPlayer && board[2][2] == currentPlayer) return 1;
    if (board[0][2] == currentPlayer && board[1][1] == currentPlayer && board[2][0] == currentPlayer) return 1;
    return 0;
}

void switchPlayer() {
    currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
}

int isBoardFull() {
    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 3; j++)
            if (board[i][j] == ' ') return 0;
    return 1;
}

int main() {
    int row, col;
    initBoard();

    while (1) {
        printBoard();
        printf("Player %c, enter row and column (0-2): ", currentPlayer);
        scanf("%d %d", &row, &col);

        if (row < 0 || row > 2 || col < 0 || col > 2 || board[row][col] != ' ') {
            printf("Invalid move, try again.\n");
            continue;
        }

        board[row][col] = currentPlayer;
```

```
        if (checkWin()) {
            printBoard();
            printf("Player %c wins!\n", currentPlayer);
            break;
        }

        if (isBoardFull()) {
            printBoard();
            printf("It's a draw!\n");
            break;
        }

        switchPlayer();
    }

    return 0;
}
```

OUTPUT:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    COMMENTS

$ java m
Current board:
- - -
- - -
- - -
Player X, enter your move (row and column):
1 1
Current board:
- - -
- X -
- - -
Player O, enter your move (row and column):
0 1
Current board:
- O -
- X -
- - -
Player X, enter your move (row and column):
2 2
Current board:
- O -
- X -
- - X
Player O, enter your move (row and column):
1 2
Current board:
- O -
- X O
- - X
Player X, enter your move (row and column):
0 0
Player X wins!
X O -
- X O
- - X

Abdurrahman Qureshi@GhouledGadget MINGW64 /d/Degree/SEM 3/PCPF/Programs (master)
```