unacademy

Function Overloading - Part I

Comprehensive Course On C++

Pankaj Sharma • Lesson 6 • May 17, 2024

unacademy

CS & IT Engineering

C++

Function Overloading

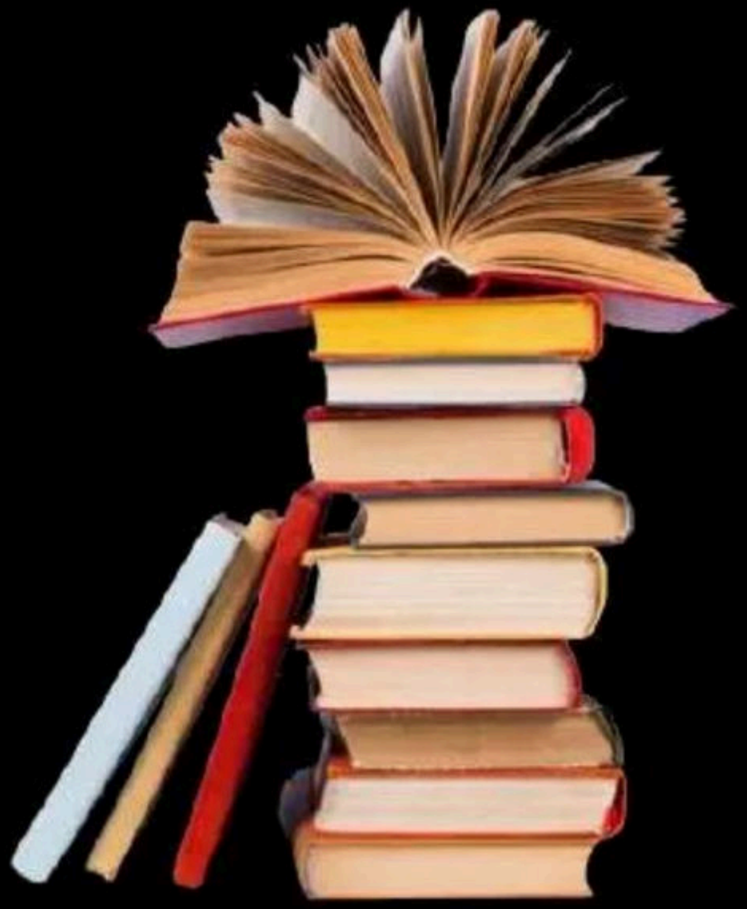
Lecture - 06

By- Pankaj Sir

unacademy

Topics

to be covered



1. Function Overloading

unacademy

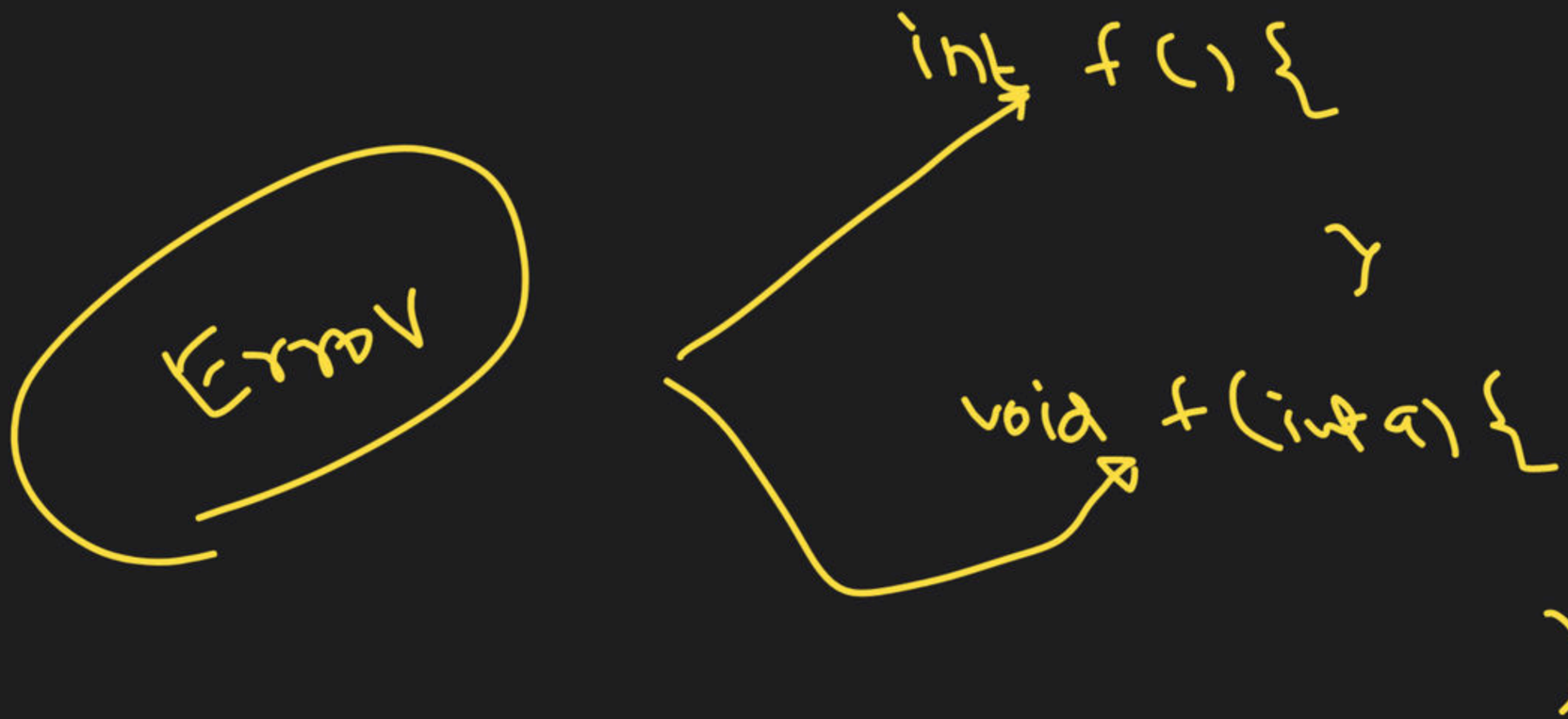
Func. Overloading

```
int FindMax(int a, int b){
    return a>b?a:b; }

int FindMax(int a, int b, int c)
{
    int max = a;
    if (b > max)
        max = b;
    if (c > max)
        max = c;
    return max;
}

int main(){
    int x=10, y=20, z=30;
    cout<< FindMax(x,y)<<" ";
    cout<< FindMax(x,y,z)<<" ";
    return 0;
}
```

C lang. ⇒ All functions must be unique.

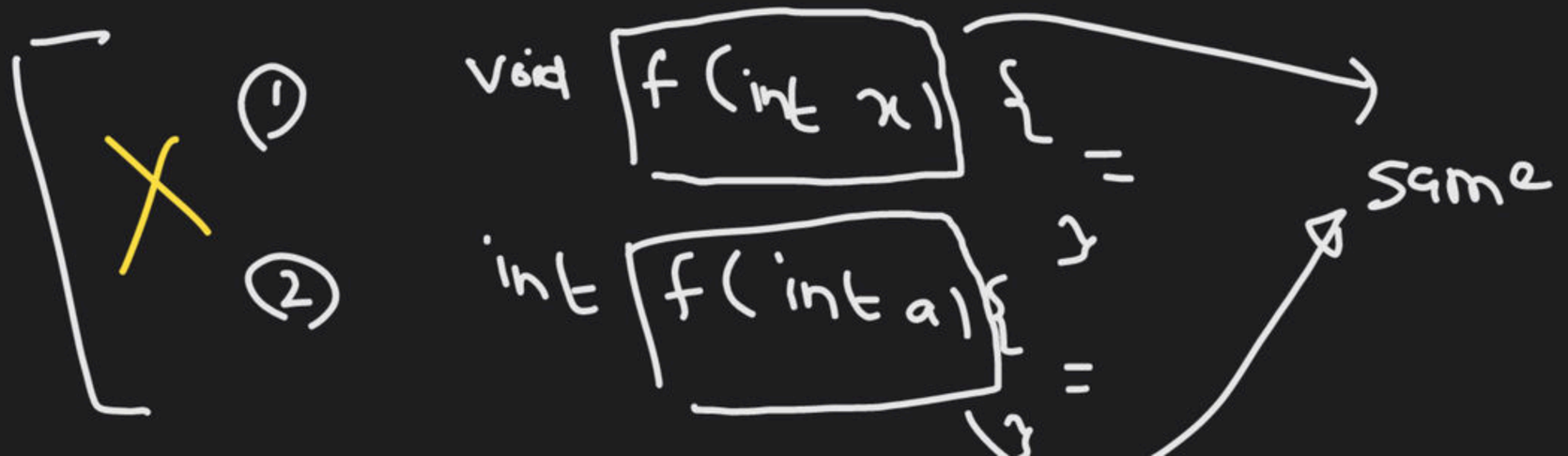


unacademy

return_type f_name (argument_list)

signature

Function signature must be unique.



unacademy

Either no. of arg. is diff

OR

type of arg. ⇒ different

unacademy

same no. of arguments

```
#include<iostream>
using namespace std;
int Add(int a, int b) { return a+b; }
double Add(double x, double y) { return x+y; }
int main(){
    int p=10, q=20;
    double r=3.2, s=4.6;
    cout<<"on integers"<<" "<<Add(p,q)<<" ";
    cout<<"on double"<<" "<<Add(r,s)<<" ";
```

unacademy

```
#include <iostream>
using namespace std;
int Area( int l, int b) { return l*b; }
int Area (int s) { return s*s;}
int main() {
    cout<<"Area of rect "<<Area(10,20)<<endl;
    cout<<"Area of sq. "<<Area(10)<<endl;
}
```

unacademy

1.) Same function name can be used in multiple definition.

2.) Function with same name must differ in argument list.

3.) We can not overload functions with same signature but different return value.

int Add (int, int);

double Add (int, int);

unacademy

t.0

Overload resolution :

unacademy

4.) Function selection (overload resolution) is performed by compiler at compile time.

5.) Function selection is based on type of arguments and no. of arguments.

unacademy

1. int f();
2. int f(int);
3. int g(int);
4. int g(double);
5. int g(char, int);
6. void h(int, char);
7. int h(int, char*);
8. int f(char, char*);
9. void f(double, double = 6.2);

```
int main(){
    f(13.2);
}
```

(1) Candidate function (by name): 1, 2, 8, 9

unacademy

int f(); ✓ ✗

int f(int); ✓ ✓

int main(){ → 1 arg

f(13.2);

}

(by name)

① Candidate function: 1, 2, 8, 9

② viable functions: (No. of arg.)

2, 9

8. int f(char, char*); ✗

9. void f(double, double = 6.2); ✓ → 1/2

unacademy

int main(){
f(13.2);
}

1 arg
double

On the basis of Exact Match

(by name)

① Candidate function: 1, 2, 8, 9
② viable functions: (No. of arg.)
1/2 → 2, 9
③ best :

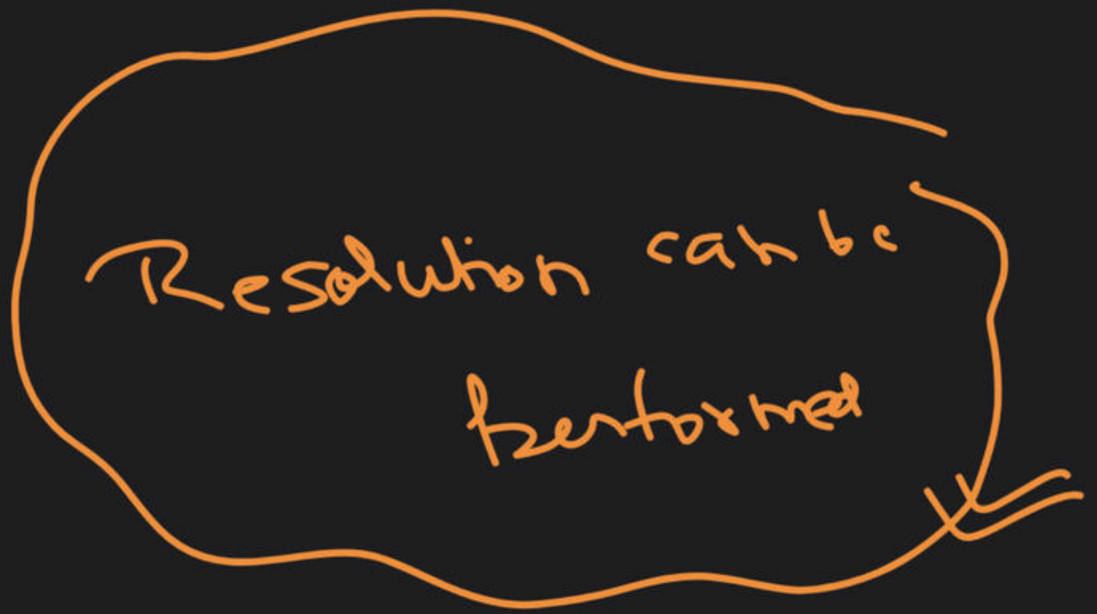
(i) Exact Match
(ii) Type Promotion
(iii) Type Conversion

int f(int); ✓ ✓

q. void f(double, double = 6.2); ✓ ✓

unacademy

```
int f (double);
```




```
void f ( double, double = 7.81);
```

unacademy

Ex 1

```
void fun(int);
int fun(float);
int main(){
    int x = 5;
    fun(x);
}
```

① Exact match

unacademy

Ex2:

```
void fun(int);
int fun(float);
int main(){
    char a = 'A';
    fun(a);
}
```

char → int

Type promotion

① Exact Match X

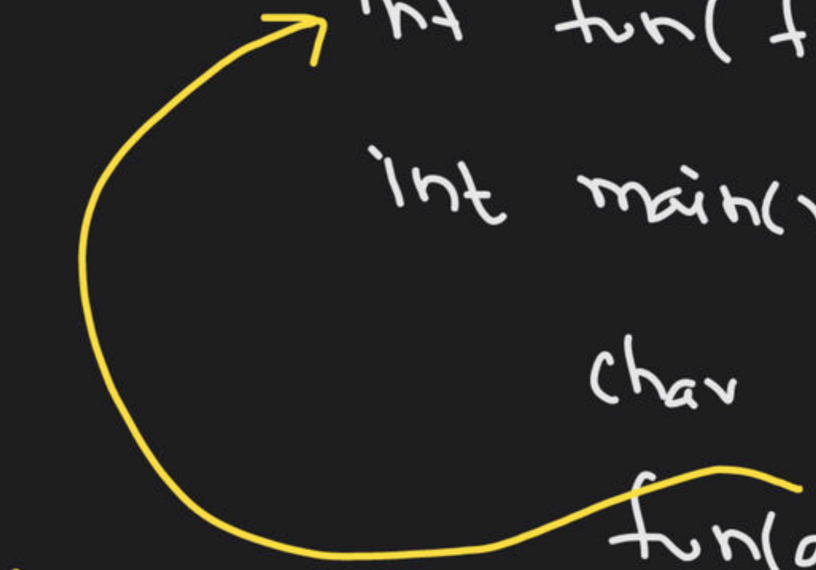
⇒ ② Type promotion:

- char to int
- float to double
- bool to int

unacademy

Ex3.

```
Void fun(struct student);
int fun(float);
int main() {
    char a = 'A';
    fun(a);
}
```

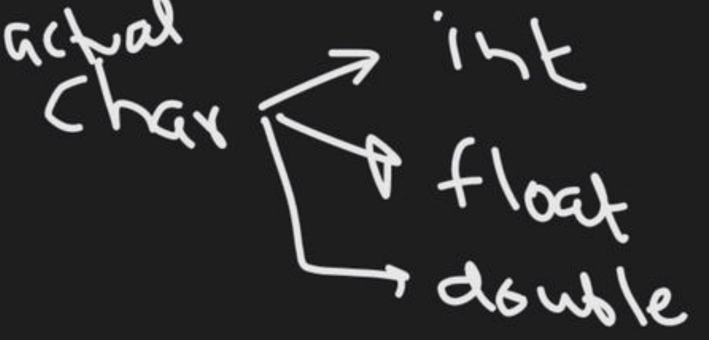


big of type Conversion =

① Exact match X

② Type promotion X

③ Type Conversion



unacademy

Ex-4

~ void fun(float);
~ int fun(double);

int main() {

char a = 'A';
fun(a);

=

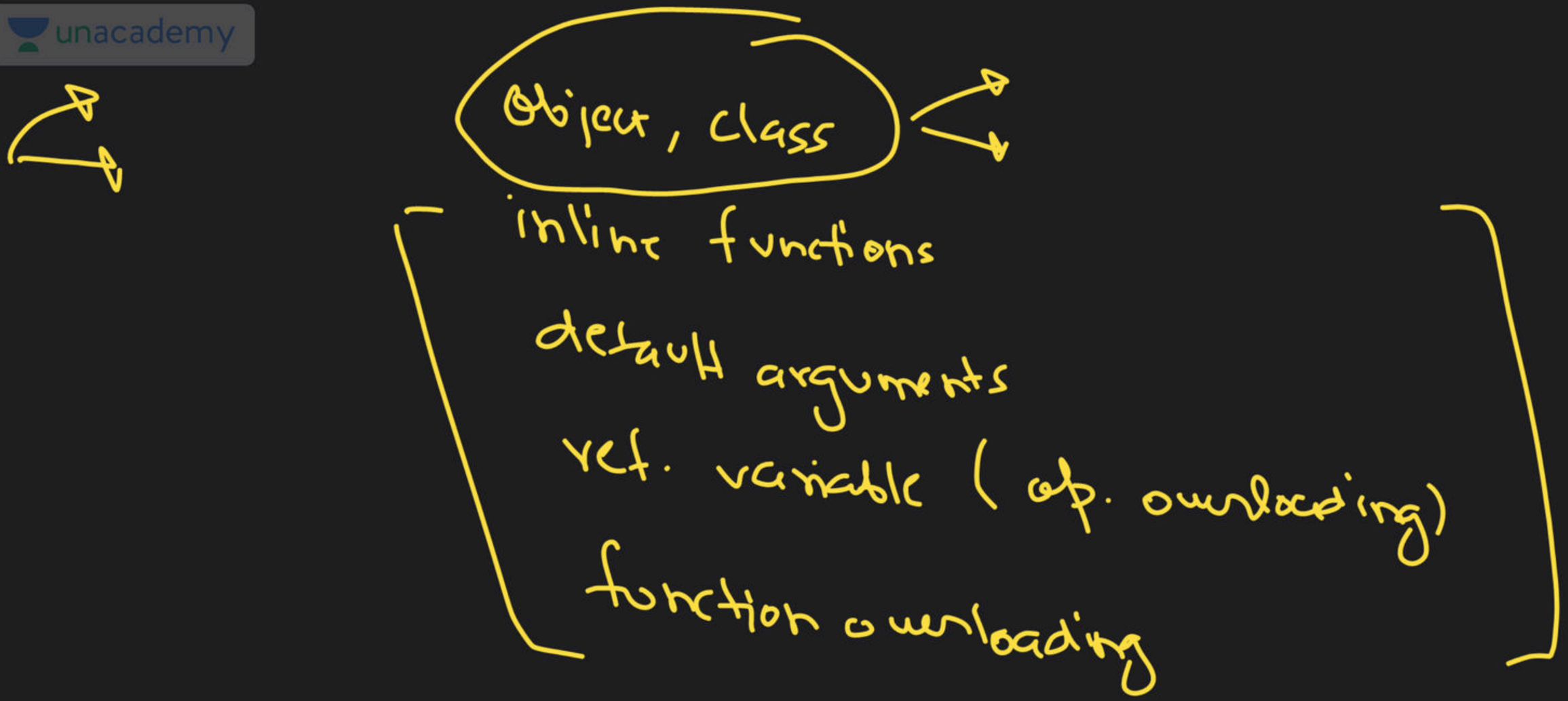
}

Error

✗ Exact Match
✗ Type Promo.
Type Conversion

char → int
→ float
→ double

unacademy


Object, Class
inline functions
default arguments
ref. variable (op. overloading)
function overloading

unacademy

unacademy

THANK YOU!

Here’s to a cracking journey ahead!