

Taazaa Training

“Assignment - 3”

Topics:-

- **Boxing and Unboxing**
- **Interface**
- **Abstract class**
- **Value type vs reference type**
- **Jagged array program**
- **2-D Array Program**

Submitted by: -

Gurpreet Singh

Boxing and Unboxing

Boxing:-

- Boxing is the process of converting a value type to the type object or to any interface type implemented by this value type.
- It is used to store value types in the heap memory.

Unboxing:-

- Unboxing is an explicit conversion from the type object to a value type or from an interface type to a value type that implements the interface. An unboxing operation consists of:
 - Checking the object instance to make sure that it is a boxed value of the given value type.
 - Copying the value from the instance into the value-type variable.
- It is used to store object types in the stack memory.

Source Code: -

```
using System;
```

```
namespace boxing_unboxing
```

```
{
```

```
    class Program
```

```
    {
```

```
        static void Main(string[] args)
```

```
        {
```

```
            //For Boxing Process
```

```
            String name="Gurpreet Singh";//(value type)
```

```
            // variable name of string type which is stored in stack memory.
```

```
            object box=name;//(reference type)
```

```
            //here object box convert value type to object type hence boxing is done
```

```
            name="Gurpreet";// new value for name variable
```

```
            Console.WriteLine("value of name before unboxing :-"+name);
```

```
            // for unboxing process
```

```
            name=(string)box;// here object type is explicitly converted to string type i.e. value type
```

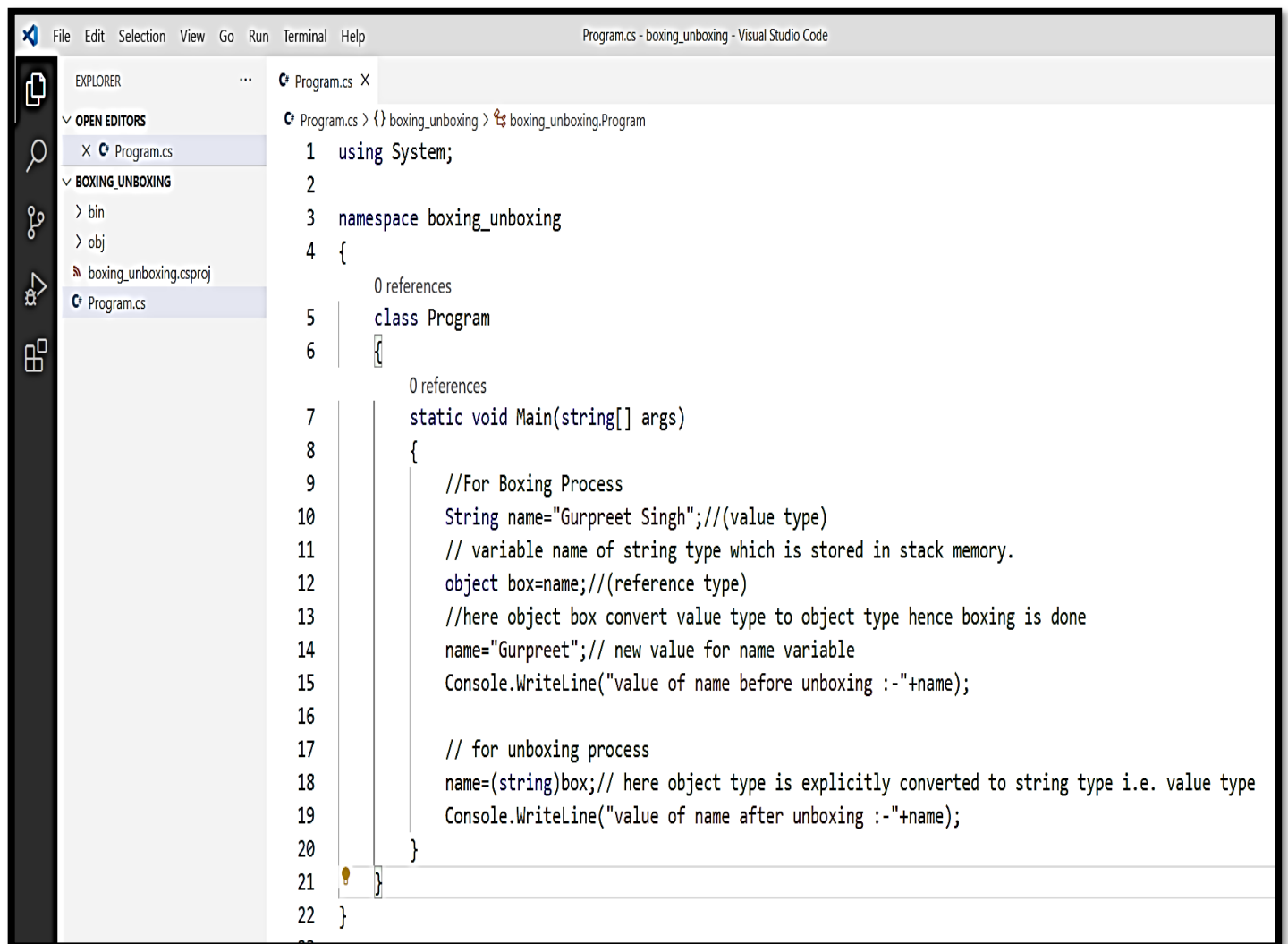
```
            Console.WriteLine("value of name after unboxing :-"+name);
```

```
        }
```

```
    }
```

```
}
```

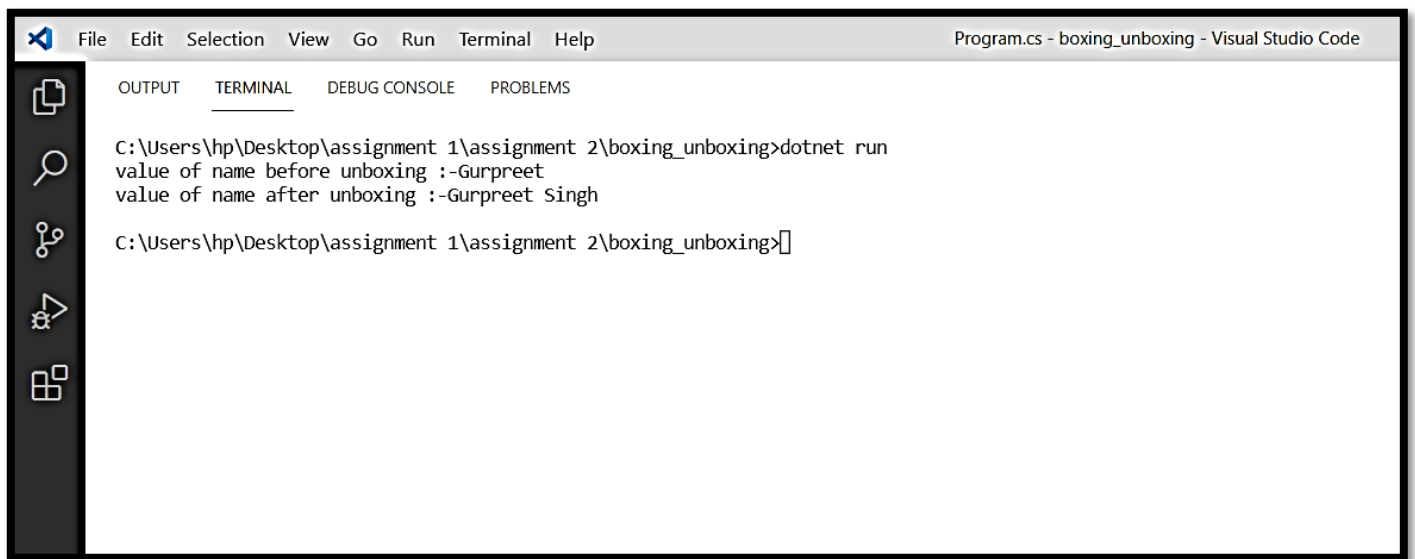
Screen Shorts:-



The screenshot displays the Visual Studio Code interface with a C# file named `Program.cs` open. The Explorer sidebar on the left shows the project structure, including the `BOXING_UNBOXING` folder and the `Program.cs` file. The main editor area shows the following code:

```
1 using System;
2
3 namespace boxing_unboxing
4 {
5     0 references
6     class Program
7     {
8         0 references
9         static void Main(string[] args)
10        {
11            //For Boxing Process
12            String name="Gurpreet Singh";//(value type)
13            // variable name of string type which is stored in stack memory.
14            object box=name;//(reference type)
15            //here object box convert value type to object type hence boxing is done
16            name="Gurpreet";// new value for name variable
17            Console.WriteLine("value of name before unboxing :-"+name);
18
19            // for unboxing process
20            name=(string)box;// here object type is explicitly converted to string type i.e. value type
21            Console.WriteLine("value of name after unboxing :-"+name);
22        }
23    }
```

Output :-

A screenshot of the Visual Studio Code interface showing the terminal output. The terminal window is titled "Program.cs - boxing_unboxing - Visual Studio Code". The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The terminal panel has tabs for OUTPUT, TERMINAL, DEBUG CONSOLE, and PROBLEMS, with TERMINAL selected. The terminal shows the execution of a .NET program. The prompt is "C:\Users\hp\Desktop\assignment 1\assignment 2\boxing_unboxing>". The first command is "dotnet run", which produces two lines of output: "value of name before unboxing :-Gurpreet" and "value of name after unboxing :-Gurpreet Singh". The second command is a prompt character ">".

```
File Edit Selection View Go Run Terminal Help Program.cs - boxing_unboxing - Visual Studio Code

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

C:\Users\hp\Desktop\assignment 1\assignment 2\boxing_unboxing>dotnet run
value of name before unboxing :-Gurpreet
value of name after unboxing :-Gurpreet Singh

C:\Users\hp\Desktop\assignment 1\assignment 2\boxing_unboxing>
```

Interface

Source Code: -

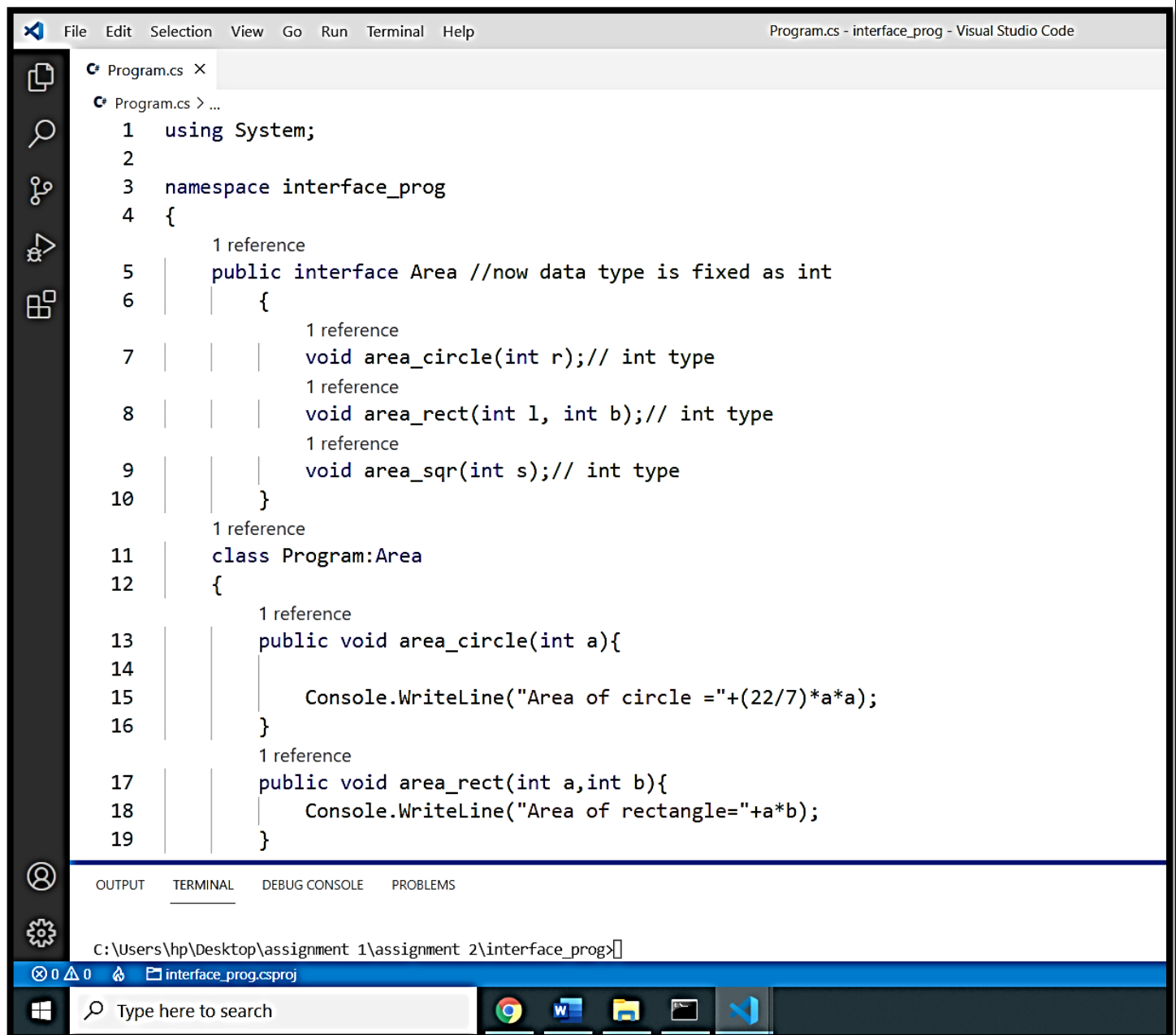
```
using System;

namespace interface_prog
{
    public interface Area //now data type is fixed as int
    {
        void area_circle(int r); // int type
        void area_rect(int l, int b); // int type
        void area_sqr(int s); // int type
    }
    class Program:Area
    {
        public void area_circle(int a){

            Console.WriteLine("Area of circle =" + (22/7)*a*a);
        }
        public void area_rect(int a, int b){
            Console.WriteLine("Area of rectangle=" + a*b);
        }
        public void area_sqr(int a){
            Console.WriteLine("Area of square=" + a*a);
        }

        static void Main(string[] args)
        {
            var obj = new Program();
            obj.area_circle(10);
            obj.area_rect(10, 20);
            obj.area_sqr(23);
        }
    }
}
```

Screen Shorts:-



The screenshot displays the Visual Studio Code editor with a C# file named `Program.cs`. The code defines an interface `Area` and a class `Program:Area` that implements it. The interface `Area` has three methods: `area_circle`, `area_rect`, and `area_sqr`, all returning `int`. The class `Program:Area` implements these methods using `Console.WriteLine` to output the results. The terminal shows the current directory path: `C:\Users\hp\Desktop\assignment 1\assignment 2\interface_prog>`.

```
1  using System;
2
3  namespace interface_prog
4  {
5      1 reference
6      public interface Area //now data type is fixed as int
7      {
8          1 reference
9          void area_circle(int r); // int type
10         1 reference
11         void area_rect(int l, int b); // int type
12         1 reference
13         void area_sqr(int s); // int type
14     }
15     1 reference
16     class Program:Area
17     {
18         1 reference
19         public void area_circle(int a){
20             Console.WriteLine("Area of circle =" + (22/7)*a*a);
21         }
22         1 reference
23         public void area_rect(int a, int b){
24             Console.WriteLine("Area of rectangle=" + a*b);
25         }
26     }
27 }
```

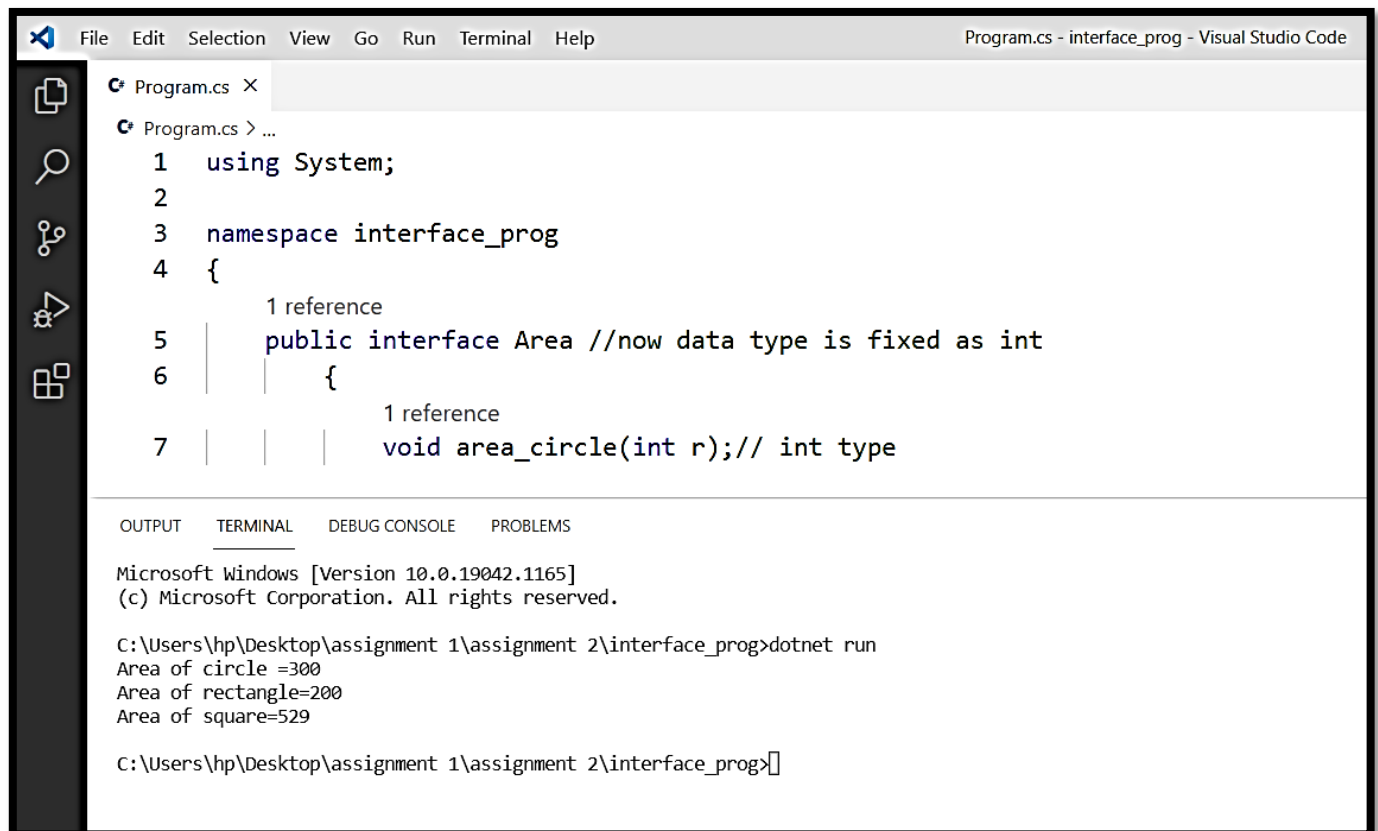
OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

C:\Users\hp\Desktop\assignment 1\assignment 2\interface_prog>

0 0 0 interface_prog.csproj

Type here to search

Output :-



The screenshot displays the Visual Studio Code interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The title bar reads "Program.cs - interface_prog - Visual Studio Code". The Explorer sidebar on the left shows a file named "Program.cs". The main editor area contains the following C# code:

```
1 using System;
2
3 namespace interface_prog
4 {
5     1 reference
6     public interface Area //now data type is fixed as int
7     {
8         1 reference
9         void area_circle(int r); // int type
```

Below the code editor, the TERMINAL tab is active, showing the following output:

```
Microsoft Windows [Version 10.0.19042.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp\Desktop\assignment 1\assignment 2\interface_prog>dotnet run
Area of circle =300
Area of rectangle=200
Area of square=529

C:\Users\hp\Desktop\assignment 1\assignment 2\interface_prog>
```

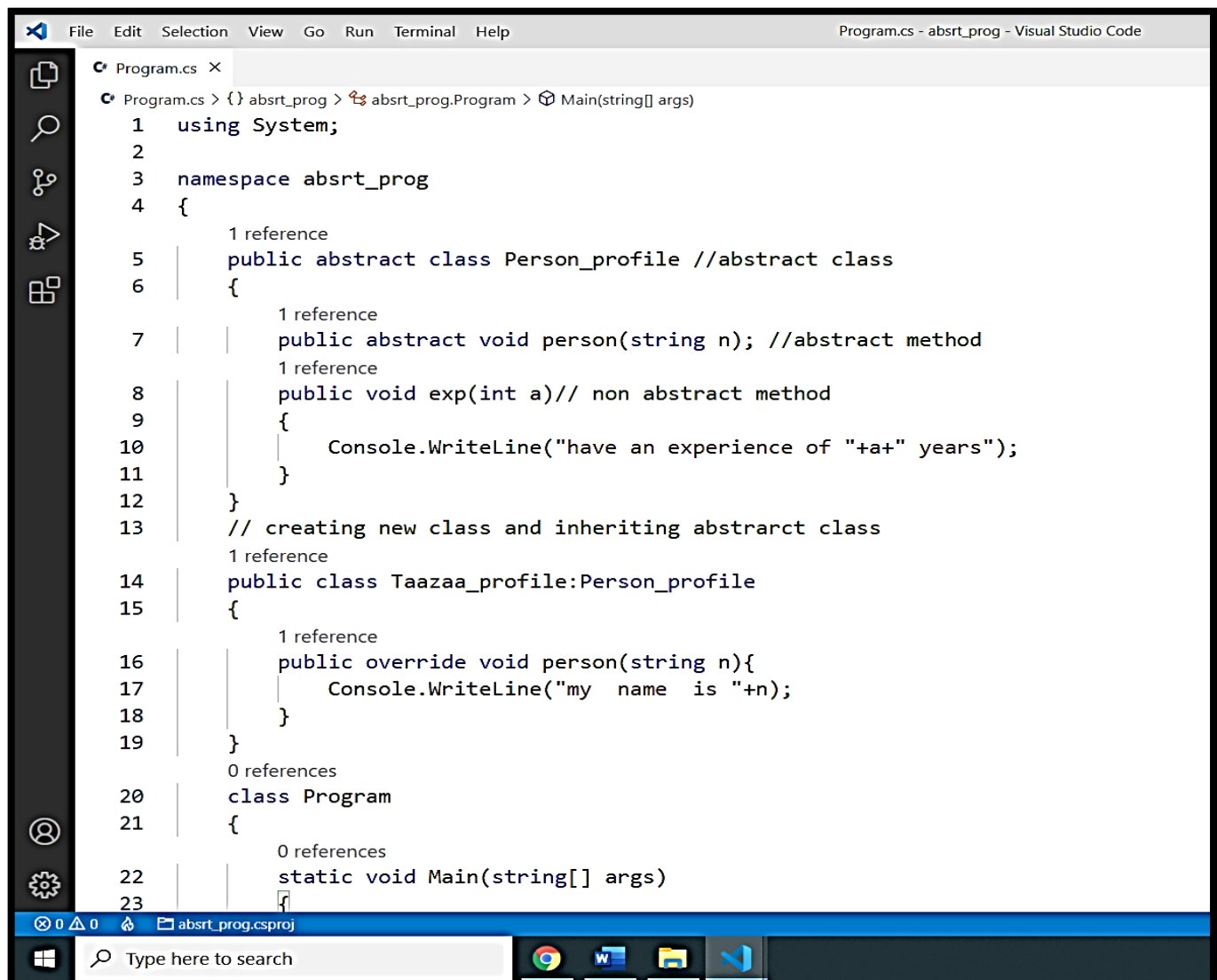
Abstract class

Source Code: -

```
using System;

namespace absrt_prog
{
    public abstract class Person_profile //abstract class
    {
        public abstract void person(string n); //abstract method
        public void exp(int a)// non abstract method
        {
            Console.WriteLine("have an experience of "+a+" years");
        }
    }
    // creating new class and inheriting abstrarct class
    public class Taazaa_profile:Person_profile
    {
        public override void person(string n){
            Console.WriteLine("my name is "+n);
        }
    }
    class Program
    {
        static void Main(string[] args)
        {
            var obj=new Taazaa_profile();
            obj.person("Gurpreet");
            obj.exp(4);
        }
    }
}
```


Screen Shorts:-

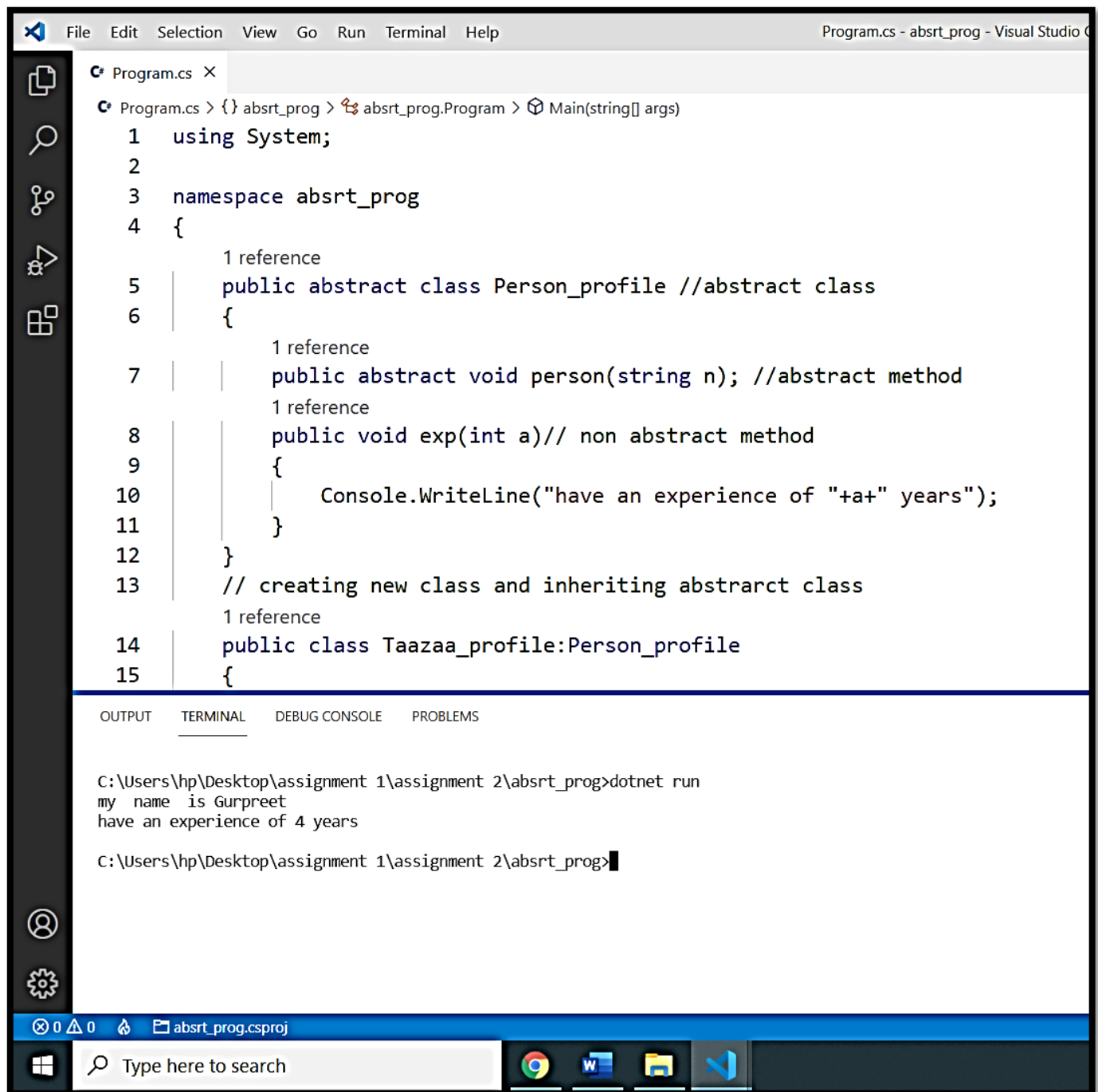


The screenshot displays the Visual Studio Code editor interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The title bar indicates the file is 'Program.cs - absrt_prog - Visual Studio Code'. The left sidebar shows the Explorer, Search, Source Control, and Run and Debug views. The main editor area shows the following C# code:

```
Program.cs > { } absrt_prog > absrt_prog.Program > Main(string[] args)
1  using System;
2
3  namespace absrt_prog
4  {
5      1 reference
6      public abstract class Person_profile //abstract class
7      {
8          1 reference
9          public abstract void person(string n); //abstract method
10         1 reference
11         public void exp(int a)// non abstract method
12         {
13             Console.WriteLine("have an experience of "+a+" years");
14         }
15     }
16     // creating new class and inheriting abstarct class
17     1 reference
18     public class Taazaa_profile:Person_profile
19     {
20         1 reference
21         public override void person(string n){
22             Console.WriteLine("my name is "+n);
23         }
24     }
25     0 references
26     class Program
27     {
28         0 references
29         static void Main(string[] args)
30         {
31         }
32     }
33 }
```

The bottom status bar shows 0 errors, 0 warnings, and 0 info messages. The taskbar at the bottom includes the Windows Start button, a search bar, and icons for Chrome, Word, File Explorer, and Visual Studio Code.

Output :-



The screenshot displays the Visual Studio IDE with a C# program. The code defines an abstract class `Person_profile` and a concrete class `Taazaa_profile` that inherits from it. The terminal shows the execution of the program, which prints the name and experience of the user.

```
File Edit Selection View Go Run Terminal Help Program.cs - absrt_prog - Visual Studio C#  
Program.cs X  
Program.cs > {} absrt_prog > absrt_prog.Program > Main(string[] args)  
1 using System;  
2  
3 namespace absrt_prog  
4 {  
5     1 reference  
6     public abstract class Person_profile //abstract class  
7     {  
8         1 reference  
9         public abstract void person(string n); //abstract method  
10        1 reference  
11        public void exp(int a)// non abstract method  
12        {  
13            Console.WriteLine("have an experience of "+a+" years");  
14        }  
15    }  
16    // creating new class and inheriting abstrarct class  
17    1 reference  
18    public class Taazaa_profile:Person_profile  
19    {  
20    }  
21    }  
22    }  
23    }  
24    }  
25    }  
26    }  
27    }  
28    }  
29    }  
30    }  
31    }  
32    }  
33    }  
34    }  
35    }  
36    }  
37    }  
38    }  
39    }  
40    }  
41    }  
42    }  
43    }  
44    }  
45    }  
46    }  
47    }  
48    }  
49    }  
50    }  
51    }  
52    }  
53    }  
54    }  
55    }  
56    }  
57    }  
58    }  
59    }  
60    }  
61    }  
62    }  
63    }  
64    }  
65    }  
66    }  
67    }  
68    }  
69    }  
70    }  
71    }  
72    }  
73    }  
74    }  
75    }  
76    }  
77    }  
78    }  
79    }  
80    }  
81    }  
82    }  
83    }  
84    }  
85    }  
86    }  
87    }  
88    }  
89    }  
90    }  
91    }  
92    }  
93    }  
94    }  
95    }  
96    }  
97    }  
98    }  
99    }  
100   }  
101   }  
102   }  
103   }  
104   }  
105   }  
106   }  
107   }  
108   }  
109   }  
110   }  
111   }  
112   }  
113   }  
114   }  
115   }  
116   }  
117   }  
118   }  
119   }  
120   }  
121   }  
122   }  
123   }  
124   }  
125   }  
126   }  
127   }  
128   }  
129   }  
130   }  
131   }  
132   }  
133   }  
134   }  
135   }  
136   }  
137   }  
138   }  
139   }  
140   }  
141   }  
142   }  
143   }  
144   }  
145   }  
146   }  
147   }  
148   }  
149   }  
150   }  
151   }  
152   }  
153   }  
154   }  
155   }  
156   }  
157   }  
158   }  
159   }  
160   }  
161   }  
162   }  
163   }  
164   }  
165   }  
166   }  
167   }  
168   }  
169   }  
170   }  
171   }  
172   }  
173   }  
174   }  
175   }  
176   }  
177   }  
178   }  
179   }  
180   }  
181   }  
182   }  
183   }  
184   }  
185   }  
186   }  
187   }  
188   }  
189   }  
190   }  
191   }  
192   }  
193   }  
194   }  
195   }  
196   }  
197   }  
198   }  
199   }  
200   }  
201   }  
202   }  
203   }  
204   }  
205   }  
206   }  
207   }  
208   }  
209   }  
210   }  
211   }  
212   }  
213   }  
214   }  
215   }  
216   }  
217   }  
218   }  
219   }  
220   }  
221   }  
222   }  
223   }  
224   }  
225   }  
226   }  
227   }  
228   }  
229   }  
230   }  
231   }  
232   }  
233   }  
234   }  
235   }  
236   }  
237   }  
238   }  
239   }  
240   }  
241   }  
242   }  
243   }  
244   }  
245   }  
246   }  
247   }  
248   }  
249   }  
250   }  
251   }  
252   }  
253   }  
254   }  
255   }  
256   }  
257   }  
258   }  
259   }  
260   }  
261   }  
262   }  
263   }  
264   }  
265   }  
266   }  
267   }  
268   }  
269   }  
270   }  
271   }  
272   }  
273   }  
274   }  
275   }  
276   }  
277   }  
278   }  
279   }  
280   }  
281   }  
282   }  
283   }  
284   }  
285   }  
286   }  
287   }  
288   }  
289   }  
290   }  
291   }  
292   }  
293   }  
294   }  
295   }  
296   }  
297   }  
298   }  
299   }  
300   }  
301   }  
302   }  
303   }  
304   }  
305   }  
306   }  
307   }  
308   }  
309   }  
310   }  
311   }  
312   }  
313   }  
314   }  
315   }  
316   }  
317   }  
318   }  
319   }  
320   }  
321   }  
322   }  
323   }  
324   }  
325   }  
326   }  
327   }  
328   }  
329   }  
330   }  
331   }  
332   }  
333   }  
334   }  
335   }  
336   }  
337   }  
338   }  
339   }  
340   }  
341   }  
342   }  
343   }  
344   }  
345   }  
346   }  
347   }  
348   }  
349   }  
350   }  
351   }  
352   }  
353   }  
354   }  
355   }  
356   }  
357   }  
358   }  
359   }  
360   }  
361   }  
362   }  
363   }  
364   }  
365   }  
366   }  
367   }  
368   }  
369   }  
370   }  
371   }  
372   }  
373   }  
374   }  
375   }  
376   }  
377   }  
378   }  
379   }  
380   }  
381   }  
382   }  
383   }  
384   }  
385   }  
386   }  
387   }  
388   }  
389   }  
390   }  
391   }  
392   }  
393   }  
394   }  
395   }  
396   }  
397   }  
398   }  
399   }  
400   }  
401   }  
402   }  
403   }  
404   }  
405   }  
406   }  
407   }  
408   }  
409   }  
410   }  
411   }  
412   }  
413   }  
414   }  
415   }  
416   }  
417   }  
418   }  
419   }  
420   }  
421   }  
422   }  
423   }  
424   }  
425   }  
426   }  
427   }  
428   }  
429   }  
430   }  
431   }  
432   }  
433   }  
434   }  
435   }  
436   }  
437   }  
438   }  
439   }  
440   }  
441   }  
442   }  
443   }  
444   }  
445   }  
446   }  
447   }  
448   }  
449   }  
450   }  
451   }  
452   }  
453   }  
454   }  
455   }  
456   }  
457   }  
458   }  
459   }  
460   }  
461   }  
462   }  
463   }  
464   }  
465   }  
466   }  
467   }  
468   }  
469   }  
470   }  
471   }  
472   }  
473   }  
474   }  
475   }  
476   }  
477   }  
478   }  
479   }  
480   }  
481   }  
482   }  
483   }  
484   }  
485   }  
486   }  
487   }  
488   }  
489   }  
490   }  
491   }  
492   }  
493   }  
494   }  
495   }  
496   }  
497   }  
498   }  
499   }  
500   }  
501   }  
502   }  
503   }  
504   }  
505   }  
506   }  
507   }  
508   }  
509   }  
510   }  
511   }  
512   }  
513   }  
514   }  
515   }  
516   }  
517   }  
518   }  
519   }  
520   }  
521   }  
522   }  
523   }  
524   }  
525   }  
526   }  
527   }  
528   }  
529   }  
530   }  
531   }  
532   }  
533   }  
534   }  
535   }  
536   }  
537   }  
538   }  
539   }  
540   }  
541   }  
542   }  
543   }  
544   }  
545   }  
546   }  
547   }  
548   }  
549   }  
550   }  
551   }  
552   }  
553   }  
554   }  
555   }  
556   }  
557   }  
558   }  
559   }  
560   }  
561   }  
562   }  
563   }  
564   }  
565   }  
566   }  
567   }  
568   }  
569   }  
570   }  
571   }  
572   }  
573   }  
574   }  
575   }  
576   }  
577   }  
578   }  
579   }  
580   }  
581   }  
582   }  
583   }  
584   }  
585   }  
586   }  
587   }  
588   }  
589   }  
590   }  
591   }  
592   }  
593   }  
594   }  
595   }  
596   }  
597   }  
598   }  
599   }  
600   }  
601   }  
602   }  
603   }  
604   }  
605   }  
606   }  
607   }  
608   }  
609   }  
610   }  
611   }  
612   }  
613   }  
614   }  
615   }  
616   }  
617   }  
618   }  
619   }  
620   }  
621   }  
622   }  
623   }  
624   }  
625   }  
626   }  
627   }  
628   }  
629   }  
630   }  
631   }  
632   }  
633   }  
634   }  
635   }  
636   }  
637   }  
638   }  
639   }  
640   }  
641   }  
642   }  
643   }  
644   }  
645   }  
646   }  
647   }  
648   }  
649   }  
650   }  
651   }  
652   }  
653   }  
654   }  
655   }  
656   }  
657   }  
658   }  
659   }  
660   }  
661   }  
662   }  
663   }  
664   }  
665   }  
666   }  
667   }  
668   }  
669   }  
670   }  
671   }  
672   }  
673   }  
674   }  
675   }  
676   }  
677   }  
678   }  
679   }  
680   }  
681   }  
682   }  
683   }  
684   }  
685   }  
686   }  
687   }  
688   }  
689   }  
690   }  
691   }  
692   }  
693   }  
694   }  
695   }  
696   }  
697   }  
698   }  
699   }  
700   }  
701   }  
702   }  
703   }  
704   }  
705   }  
706   }  
707   }  
708   }  
709   }  
710   }  
711   }  
712   }  
713   }  
714   }  
715   }  
716   }  
717   }  
718   }  
719   }  
720   }  
721   }  
722   }  
723   }  
724   }  
725   }  
726   }  
727   }  
728   }  
729   }  
730   }  
731   }  
732   }  
733   }  
734   }  
735   }  
736   }  
737   }  
738   }  
739   }  
740   }  
741   }  
742   }  
743   }  
744   }  
745   }  
746   }  
747   }  
748   }  
749   }  
750   }  
751   }  
752   }  
753   }  
754   }  
755   }  
756   }  
757   }  
758   }  
759   }  
760   }  
761   }  
762   }  
763   }  
764   }  
765   }  
766   }  
767   }  
768   }  
769   }  
770   }  
771   }  
772   }  
773   }  
774   }  
775   }  
776   }  
777   }  
778   }  
779   }  
780   }  
781   }  
782   }  
783   }  
784   }  
785   }  
786   }  
787   }  
788   }  
789   }  
790   }  
791   }  
792   }  
793   }  
794   }  
795   }  
796   }  
797   }  
798   }  
799   }  
800   }  
801   }  
802   }  
803   }  
804   }  
805   }  
806   }  
807   }  
808   }  
809   }  
810   }  
811   }  
812   }  
813   }  
814   }  
815   }  
816   }  
817   }  
818   }  
819   }  
820   }  
821   }  
822   }  
823   }  
824   }  
825   }  
826   }  
827   }  
828   }  
829   }  
830   }  
831   }  
832   }  
833   }  
834   }  
835   }  
836   }  
837   }  
838   }  
839   }  
840   }  
841   }  
842   }  
843   }  
844   }  
845   }  
846   }  
847   }  
848   }  
849   }  
850   }  
851   }  
852   }  
853   }  
854   }  
855   }  
856   }  
857   }  
858   }  
859   }  
860   }  
861   }  
862   }  
863   }  
864   }  
865   }  
866   }  
867   }  
868   }  
869   }  
870   }  
871   }  
872   }  
873   }  
874   }  
875   }  
876   }  
877   }  
878   }  
879   }  
880   }  
881   }  
882   }  
883   }  
884   }  
885   }  
886   }  
887   }  
888   }  
889   }  
890   }  
891   }  
892   }  
893   }  
894   }  
895   }  
896   }  
897   }  
898   }  
899   }  
900   }  
901   }  
902   }  
903   }  
904   }  
905   }  
906   }  
907   }  
908   }  
909   }  
910   }  
911   }  
912   }  
913   }  
914   }  
915   }  
916   }  
917   }  
918   }  
919   }  
920   }  
921   }  
922   }  
923   }  
924   }  
925   }  
926   }  
927   }  
928   }  
929   }  
930   }  
931   }  
932   }  
933   }  
934   }  
935   }  
936   }  
937   }  
938   }  
939   }  
940   }  
941   }  
942   }  
943   }  
944   }  
945   }  
946   }  
947   }  
948   }  
949   }  
950   }  
951   }  
952   }  
953   }  
954   }  
955   }  
956   }  
957   }  
958   }  
959   }  
960   }  
961   }  
962   }  
963   }  
964   }  
965   }  
966   }  
967   }  
968   }  
969   }  
970   }  
971   }  
972   }  
973   }  
974   }  
975   }  
976   }  
977   }  
978   }  
979   }  
980   }  
981   }  
982   }  
983   }  
984   }  
985   }  
986   }  
987   }  
988   }  
989   }  
990   }  
991   }  
992   }  
993   }  
994   }  
995   }  
996   }  
997   }  
998   }  
999   }  
1000  }
```

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

C:\Users\hp\Desktop\assignment 1\assignment 2\absrt_prog>dotnet run
my name is Gurpreet
have an experience of 4 years
C:\Users\hp\Desktop\assignment 1\assignment 2\absrt_prog>

0 0 0 absrt_prog.csproj

Type here to search

Value type vs Reference type

Source Code: -

For Value type

```
using System;

namespace vtandrt
{
    class Program
    {
        public static void ChangeValue(int x)
        {
            x = 200;
            Console.WriteLine(x);
        }
    }

    public static void Main(string[] args)
    {
        int i = 100;

        Console.WriteLine(i);

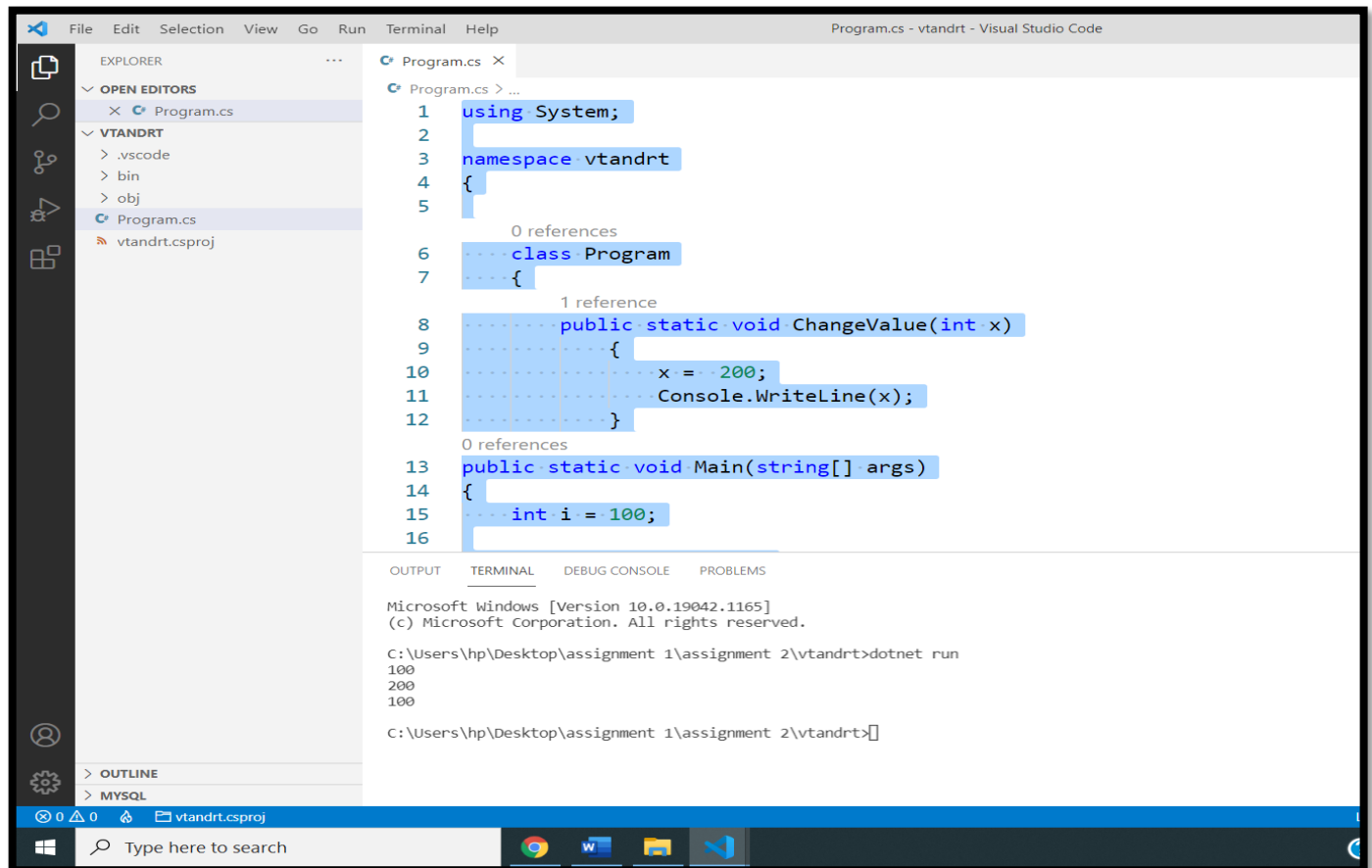
        ChangeValue(i);

        Console.WriteLine(i);
    }
}
```

For Reference type:-

Screen Shorts:-

For Value type



```
1 using System;
2
3 namespace vtandrt
4 {
5
6     0 references
7     class Program
8     {
9         1 reference
10        public static void ChangeValue(int x)
11        {
12            x = 200;
13            Console.WriteLine(x);
14        }
15
16        0 references
17        public static void Main(string[] args)
18        {
19            int i = 100;
20        }
21    }
22 }
```

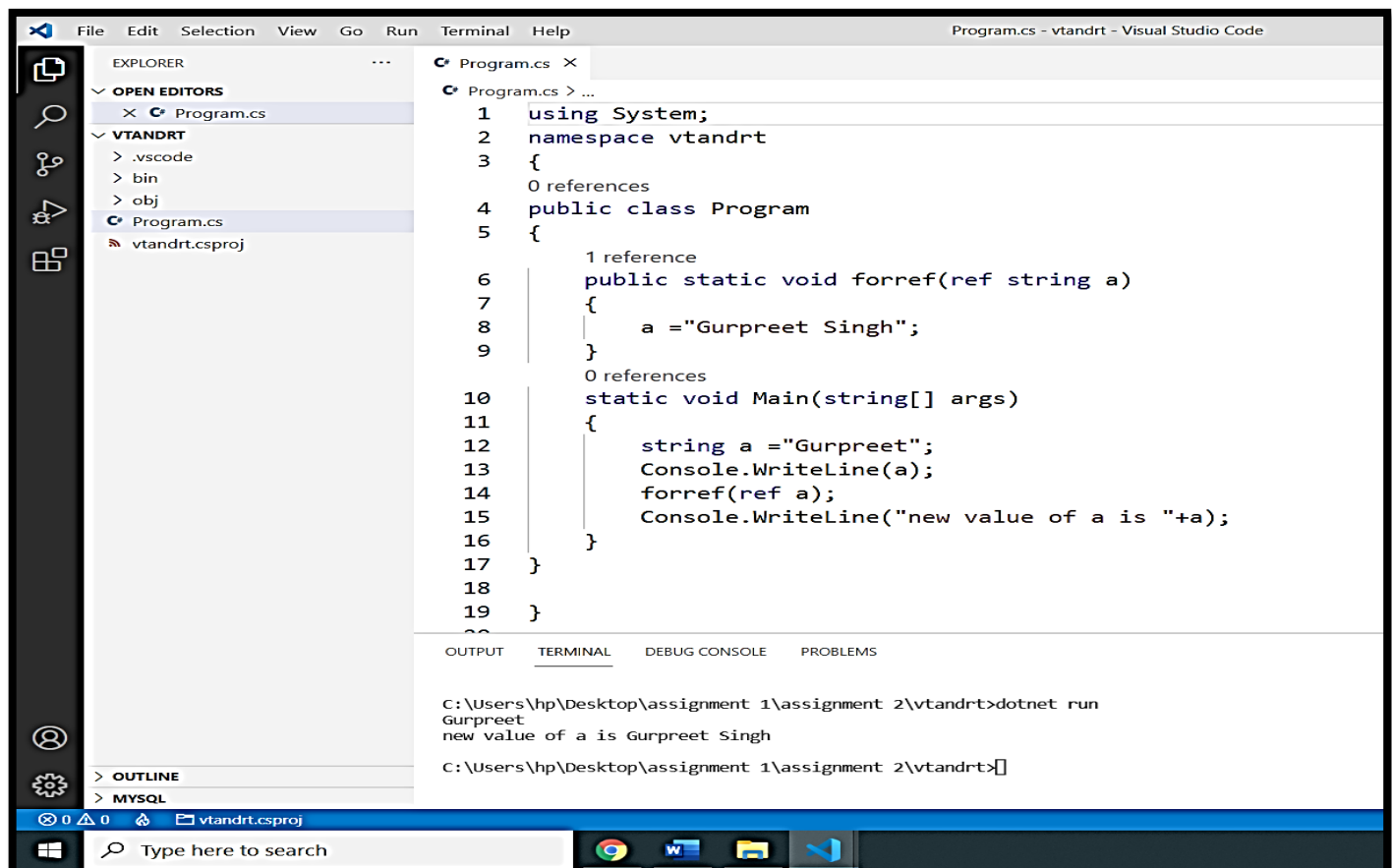
Microsoft Windows [Version 10.0.19042.1165]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp\Desktop\assignment 1\assignment 2\vtandrt>dotnet run

100
200
100

C:\Users\hp\Desktop\assignment 1\assignment 2\vtandrt>[]

For Reference type:-



```
1 using System;
2 namespace vtandrt
3 {
4     0 references
5     public class Program
6     {
7         1 reference
8         public static void forref(ref string a)
9         {
10             a = "Gurpreet Singh";
11         }
12
13         0 references
14         static void Main(string[] args)
15         {
16             string a = "Gurpreet";
17             Console.WriteLine(a);
18             forref(ref a);
19             Console.WriteLine("new value of a is "+a);
20         }
21     }
22 }
```

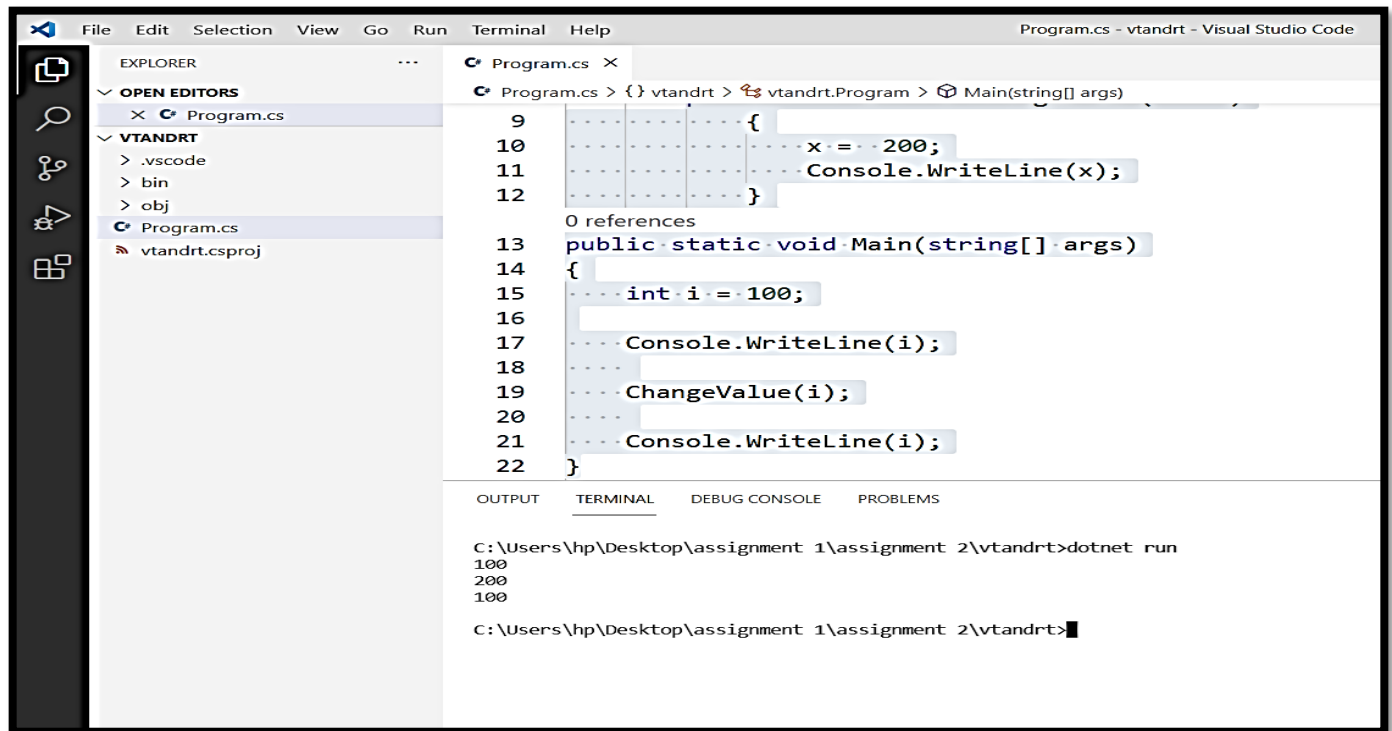
C:\Users\hp\Desktop\assignment 1\assignment 2\vtandrt>dotnet run

Gurpreet
new value of a is Gurpreet Singh

C:\Users\hp\Desktop\assignment 1\assignment 2\vtandrt>[]

Output :-

For Value type



```
File Edit Selection View Go Run Terminal Help Program.cs - vtandrt - Visual Studio Code

EXPLORER
  OPEN EDITORS
    Program.cs
  VTANDRT
    .vscode
    bin
    obj
    Program.cs
    vtandrt.csproj

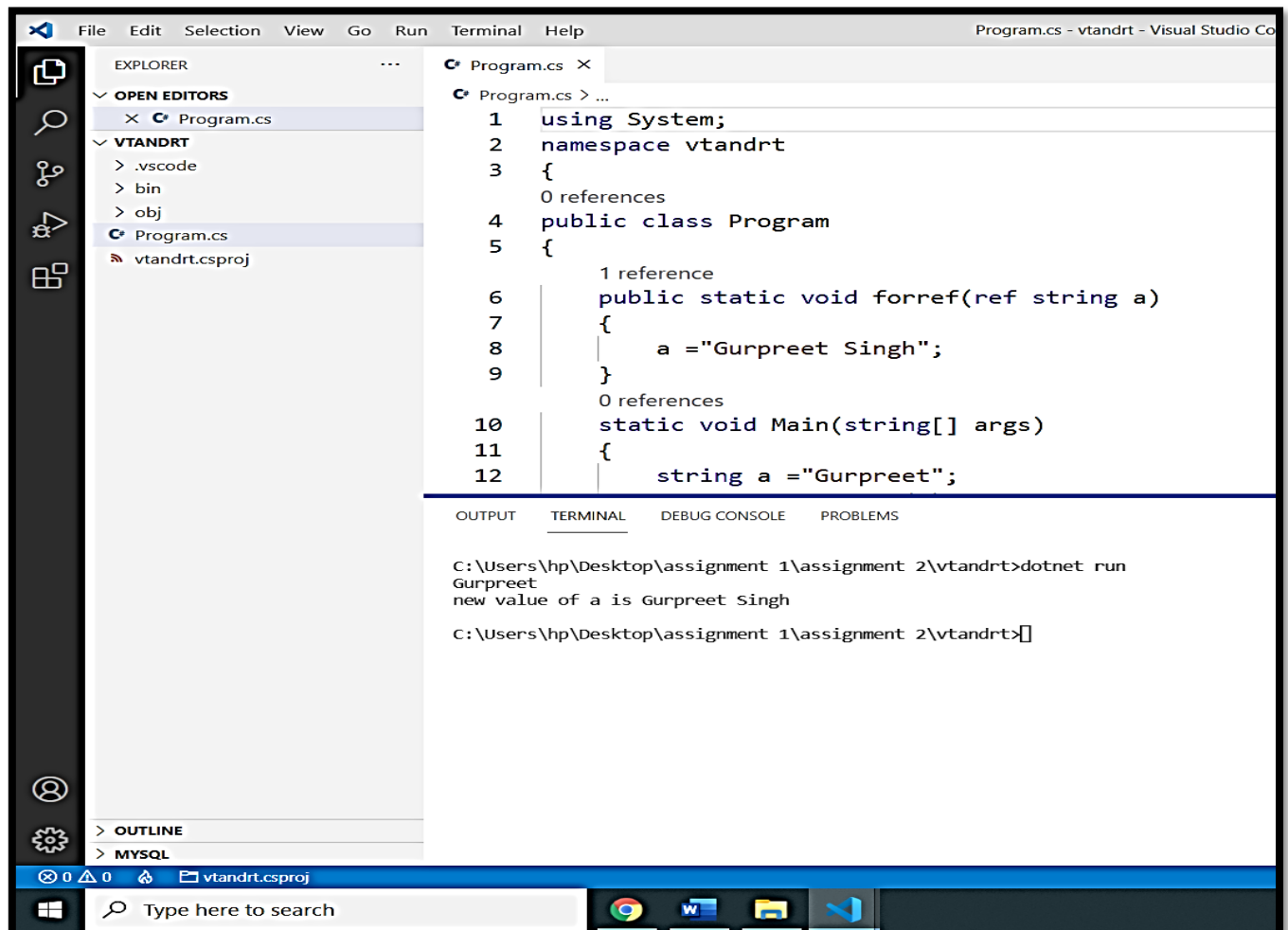
Program.cs
9  ..... {
10 ..... x = 200;
11 ..... Console.WriteLine(x);
12 ..... }
0 references
13 public static void Main(string[] args)
14 {
15   int i = 100;
16
17   Console.WriteLine(i);
18
19   ChangeValue(i);
20
21   Console.WriteLine(i);
22 }

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

C:\Users\hp\Desktop\assignment 1\assignment 2\vtandrt>dotnet run
100
200
100

C:\Users\hp\Desktop\assignment 1\assignment 2\vtandrt>
```

For Reference type:-



```
File Edit Selection View Go Run Terminal Help Program.cs - vtandrt - Visual Studio Co

EXPLORER
  OPEN EDITORS
    Program.cs
  VTANDRT
    .vscode
    bin
    obj
    Program.cs
    vtandrt.csproj

Program.cs
1  using System;
2  namespace vtandrt
3  {
4  0 references
4  public class Program
5  {
6  1 reference
6  public static void forref(ref string a)
7  {
8  a = "Gurpreet Singh";
9  }
10 0 references
10 static void Main(string[] args)
11 {
12 string a = "Gurpreet";

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

C:\Users\hp\Desktop\assignment 1\assignment 2\vtandrt>dotnet run
Gurpreet
new value of a is Gurpreet Singh

C:\Users\hp\Desktop\assignment 1\assignment 2\vtandrt>
```

Jagged array program

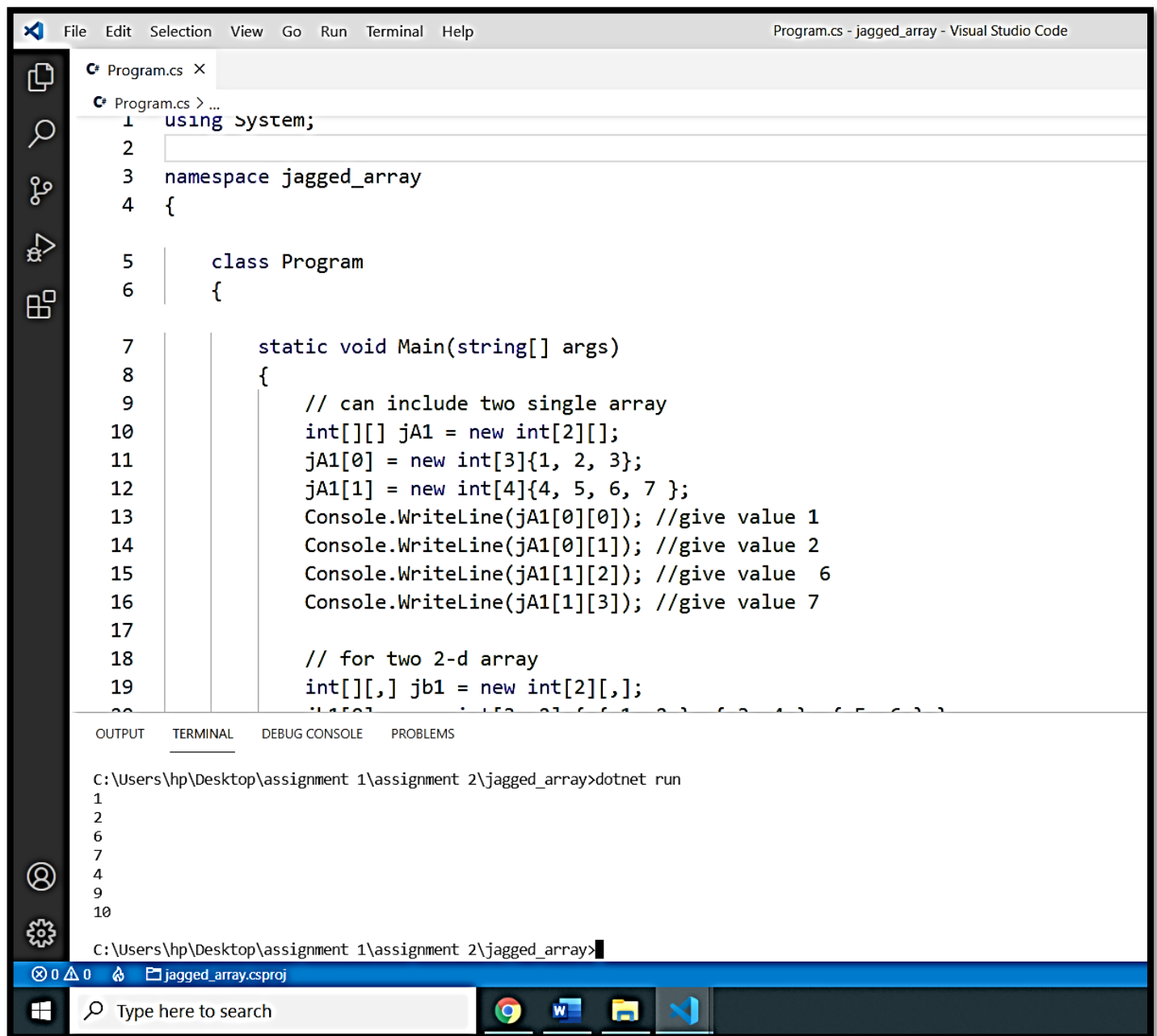
Source Code: -

```
using System;

namespace jagged_array
{
    class Program
    {
        static void Main(string[] args)
        {
            // can include two single array
            int[][] jA1 = new int[2][];
            jA1[0] = new int[3]{1, 2, 3};
            jA1[1] = new int[4]{4, 5, 6, 7 };
            Console.WriteLine(jA1[0][0]); //give value 1
            Console.WriteLine(jA1[0][1]); //give value 2
            Console.WriteLine(jA1[1][2]); //give value 6
            Console.WriteLine(jA1[1][3]); //give value 7

            // for two 2-d array
            int[,] jb1 = new int[2][,];
            jb1[0] = new int[3, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 } };
            jb1[1] = new int[2, 2] { { 7, 8 }, { 9, 10 } };
            Console.WriteLine(jb1[0][1, 1]); // give value 4
            Console.WriteLine(jb1[1][1, 0]); // give value 9
            Console.WriteLine(jb1[1][1, 1]); // give value 10
        }
    }
}
```

Screen Shorts:-



The screenshot displays the Visual Studio Code editor with a C# file named `Program.cs`. The code defines a `jagged_array` namespace and a `Program` class with a `Main` method. The `Main` method demonstrates the creation and usage of jagged arrays. It first creates a 2D array `jA1` and populates it with specific values, then prints them. Next, it creates another 2D array `jb1` and prints its values.

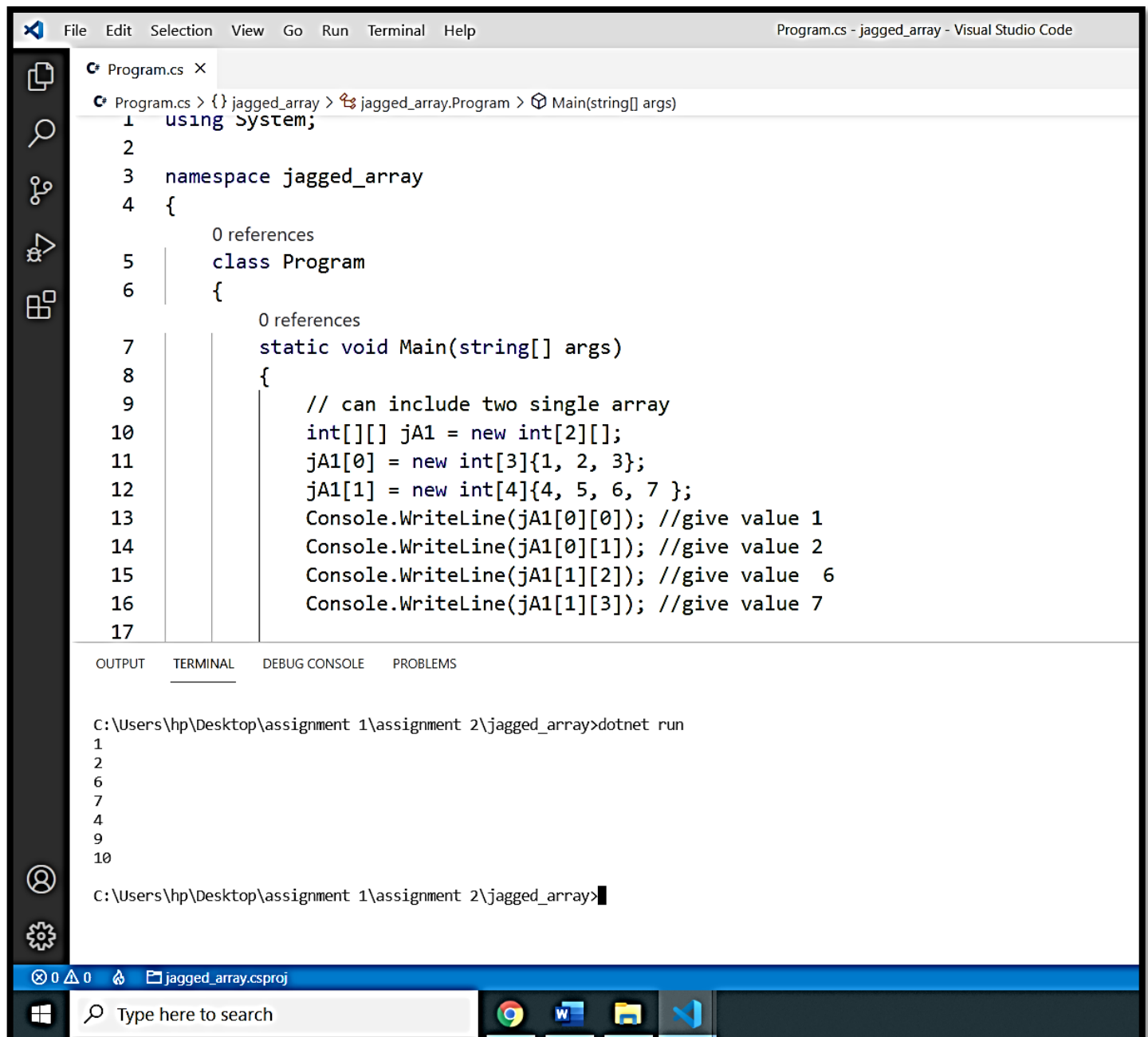
```
1 using System;
2
3 namespace jagged_array
4 {
5     class Program
6     {
7         static void Main(string[] args)
8         {
9             // can include two single array
10            int[][] jA1 = new int[2][];
11            jA1[0] = new int[3]{1, 2, 3};
12            jA1[1] = new int[4]{4, 5, 6, 7 };
13            Console.WriteLine(jA1[0][0]); //give value 1
14            Console.WriteLine(jA1[0][1]); //give value 2
15            Console.WriteLine(jA1[1][2]); //give value 6
16            Console.WriteLine(jA1[1][3]); //give value 7
17
18            // for two 2-d array
19            int[,] jb1 = new int[2][,];
20            jb1[0,0] = 1; jb1[0,1] = 2; jb1[0,2] = 3; jb1[0,3] = 4;
21            jb1[1,0] = 5; jb1[1,1] = 6; jb1[1,2] = 7; jb1[1,3] = 8;
```

The terminal window shows the command `dotnet run` being executed, resulting in the following output:

```
C:\Users\hp\Desktop\assignment 1\assignment 2\jagged_array>dotnet run
1
2
6
7
4
9
10
C:\Users\hp\Desktop\assignment 1\assignment 2\jagged_array>
```

The status bar at the bottom indicates the current file is `jagged_array.csproj`.

Output :-



The screenshot displays the Visual Studio Code interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The title bar indicates the file is 'Program.cs - jagged_array - Visual Studio Code'. The editor shows a C# file named 'Program.cs' with the following code:

```
1 using System;
2
3 namespace jagged_array
4 {
5     0 references
6     class Program
7     {
8         0 references
9         static void Main(string[] args)
10        {
11            // can include two single array
12            int[][] jA1 = new int[2][];
13            jA1[0] = new int[3]{1, 2, 3};
14            jA1[1] = new int[4]{4, 5, 6, 7 };
15            Console.WriteLine(jA1[0][0]); //give value 1
16            Console.WriteLine(jA1[0][1]); //give value 2
17            Console.WriteLine(jA1[1][2]); //give value 6
18            Console.WriteLine(jA1[1][3]); //give value 7
19        }
20    }
21 }
```

Below the editor, the 'TERMINAL' tab is active, showing the command prompt output:

```
C:\Users\hp\Desktop\assignment 1\assignment 2\jagged_array>dotnet run
1
2
6
7
4
9
10
C:\Users\hp\Desktop\assignment 1\assignment 2\jagged_array>
```

The status bar at the bottom shows '0 0 0' errors, warnings, and info, and the file 'jagged_array.csproj' is open. The Windows taskbar is visible at the very bottom with icons for the Start menu, search, and several applications including Chrome, Word, File Explorer, and VS Code.

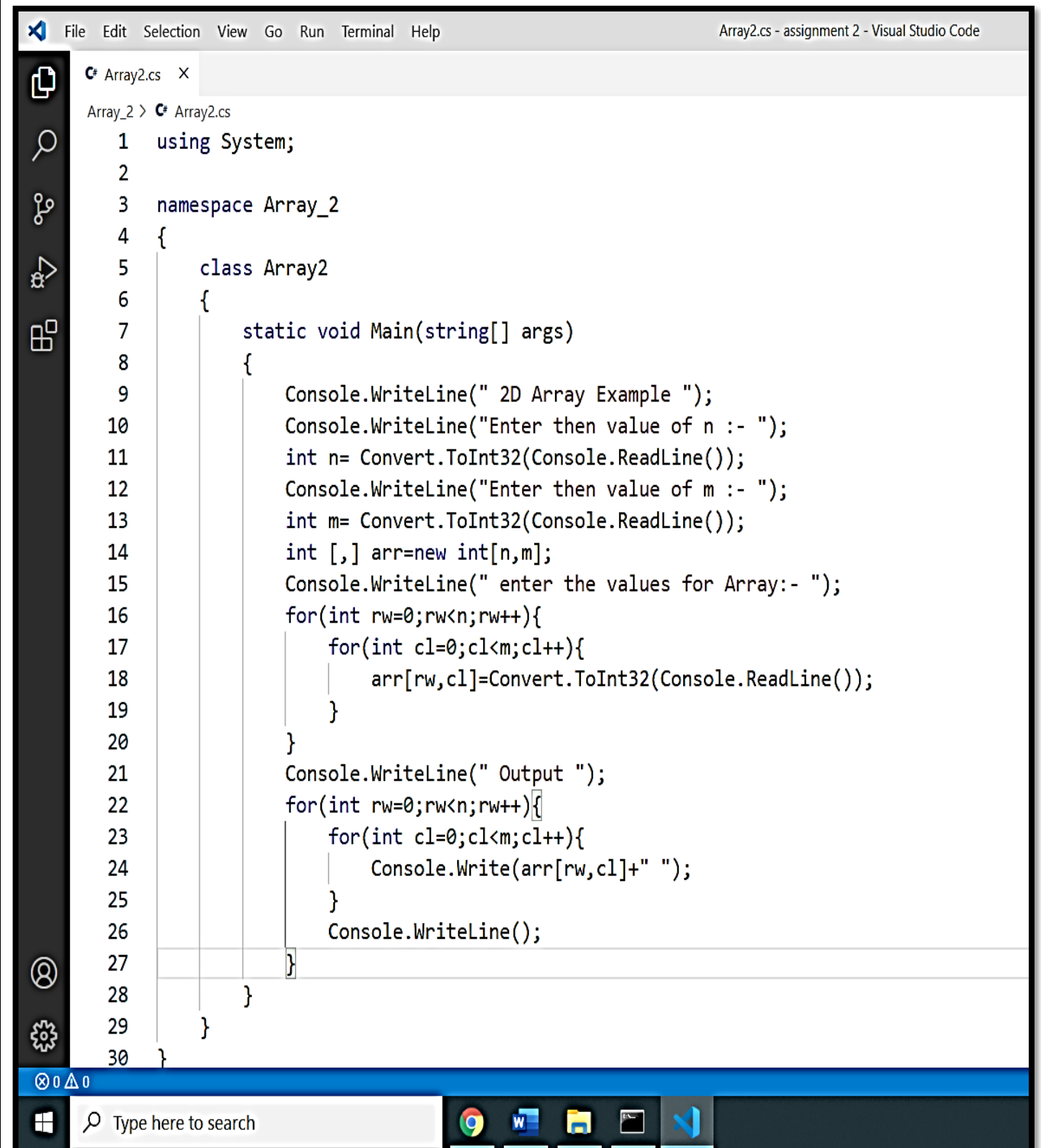
2-D Array Program

Source Code: -

```
using System;

namespace Array_2
{
    class Array2
    {
        static void Main(string[] args)
        {
            Console.WriteLine(" 2D Array Example ");
            Console.WriteLine("Enter then value of n :- ");
            int n= Convert.ToInt32(Console.ReadLine());
            Console.WriteLine("Enter then value of m :- ");
            int m= Convert.ToInt32(Console.ReadLine());
            int [,] arr=new int[n,m];
            Console.WriteLine(" enter the values for Array:- ");
            for(int rw=0;rw<n;rw++){
                for(int cl=0;cl<m;cl++){
                    arr[rw,cl]=Convert.ToInt32(Console.ReadLine());
                }
            }
            Console.WriteLine(" Output ");
            for(int rw=0;rw<n;rw++){
                for(int cl=0;cl<m;cl++){
                    Console.Write(arr[rw,cl]+" ");
                }
                Console.WriteLine();
            }
        }
    }
}
```

Screen Shorts:-

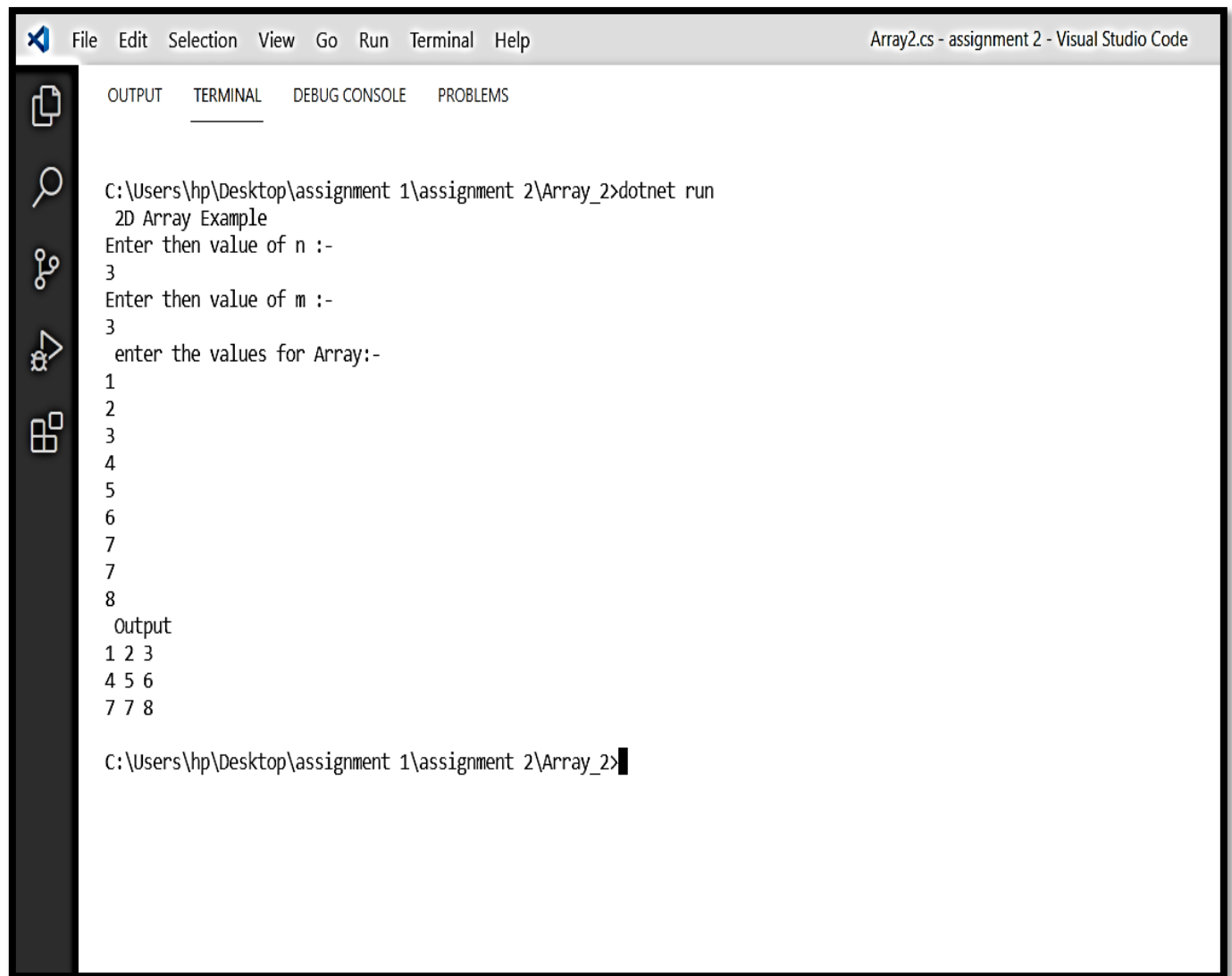


The screenshot displays the Visual Studio Code interface with a C# file named `Array2.cs` open. The code is a console application that prompts the user to enter the dimensions of a 2D array (n and m) and then prints the array's contents. The code is as follows:

```
1 using System;
2
3 namespace Array_2
4 {
5     class Array2
6     {
7         static void Main(string[] args)
8         {
9             Console.WriteLine(" 2D Array Example ");
10            Console.WriteLine("Enter then value of n :- ");
11            int n= Convert.ToInt32(Console.ReadLine());
12            Console.WriteLine("Enter then value of m :- ");
13            int m= Convert.ToInt32(Console.ReadLine());
14            int [,] arr=new int[n,m];
15            Console.WriteLine(" enter the values for Array:- ");
16            for(int rw=0;rw<n;rw++){
17                for(int cl=0;cl<m;cl++){
18                    arr[rw,cl]=Convert.ToInt32(Console.ReadLine());
19                }
20            }
21            Console.WriteLine(" Output ");
22            for(int rw=0;rw<n;rw++){
23                for(int cl=0;cl<m;cl++){
24                    Console.Write(arr[rw,cl]+" ");
25                }
26                Console.WriteLine();
27            }
28        }
29    }
30 }
```

The bottom of the image shows the Windows taskbar with the search bar and several application icons including Google Chrome, Microsoft Word, File Explorer, and the Visual Studio Code icon.

Output :-



The screenshot shows the Visual Studio Code interface with the 'TERMINAL' tab selected. The terminal window displays the execution of a C# program. The user enters '3' for 'n' and '3' for 'm'. The program then prompts for array values, and the user enters '1 2 3', '4 5 6', and '7 7 8' on three separate lines. The program outputs these values in a 3x3 grid format.

```
File Edit Selection View Go Run Terminal Help Array2.cs - assignment 2 - Visual Studio Code

OUTPUT TERMINAL DEBUG CONSOLE PROBLEMS

C:\Users\hp\Desktop\assignment 1\assignment 2\Array_2>dotnet run
2D Array Example
Enter then value of n :-
3
Enter then value of m :-
3
enter the values for Array:-
1
2
3
4
5
6
7
7
8
Output
1 2 3
4 5 6
7 7 8

C:\Users\hp\Desktop\assignment 1\assignment 2\Array_2>
```