Assignment - 8

a] What is meant by virtual address

→

Virtual address:-

- A virtual address is a binary number in virtual memory that lets a process use a location in primary storage or in some cases in secondary storage
- A virtual address space is the set of ranges of virtual addresses that an operating system makes available to a process.
- The range of virtual addresses usually starts at a low address & can extend to the highest address allowed by the computer instruction set.

Q2) what is the use of & (address of) operator?

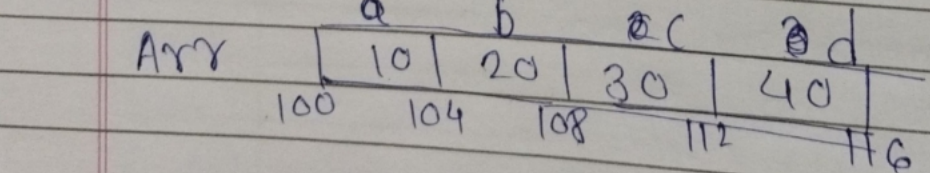
→
- & is called as address of operator
- address of operator (&) is used to fetch the Address of any data object.
- This operator returns an integer value which is the address of its operand in memory.
- we use address of (&) operator with any kind of variables, arra strings, functions & pointer.

Syntax:

```
& operand;
& a;
```

e.g: address of operator in Arra

```
int Arr [4] = {10, 20, 30, 40};
```



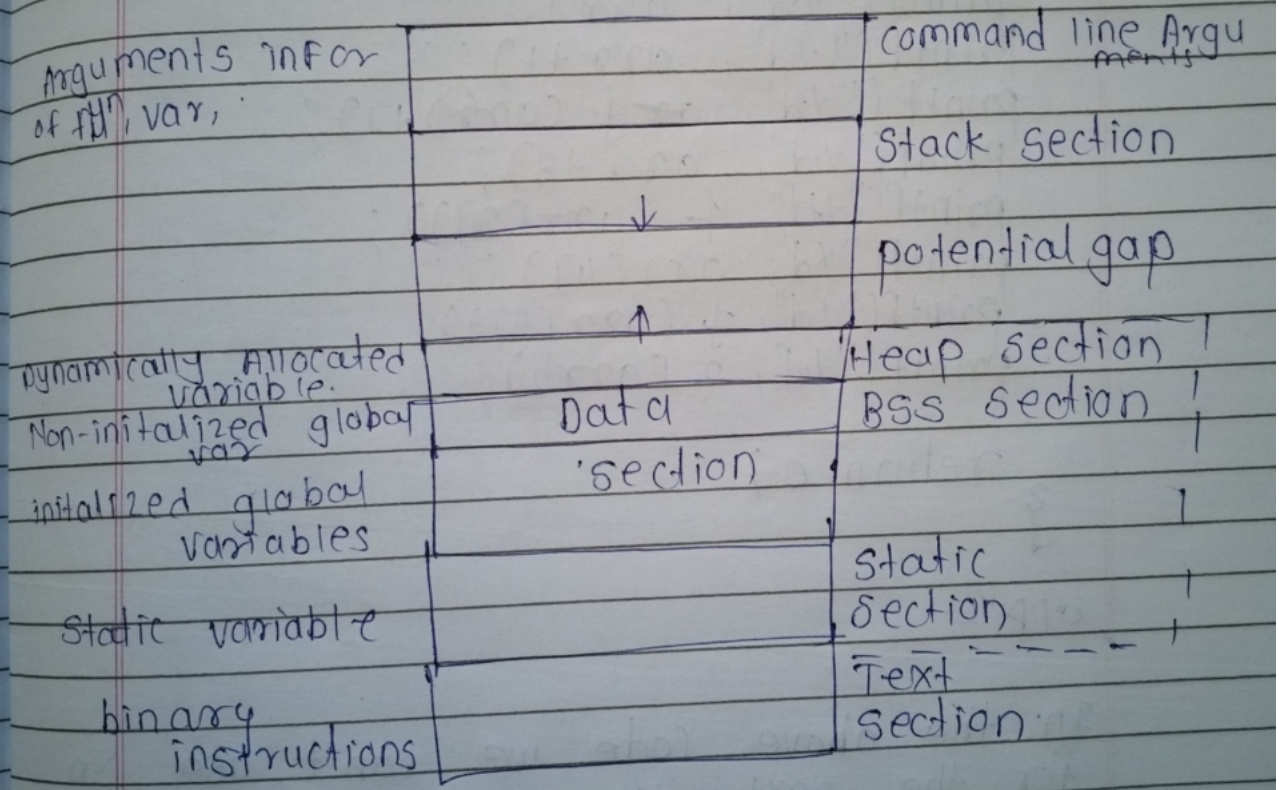

```
& a = 100;
& d = 112
```

Q3] What is Address space of process?

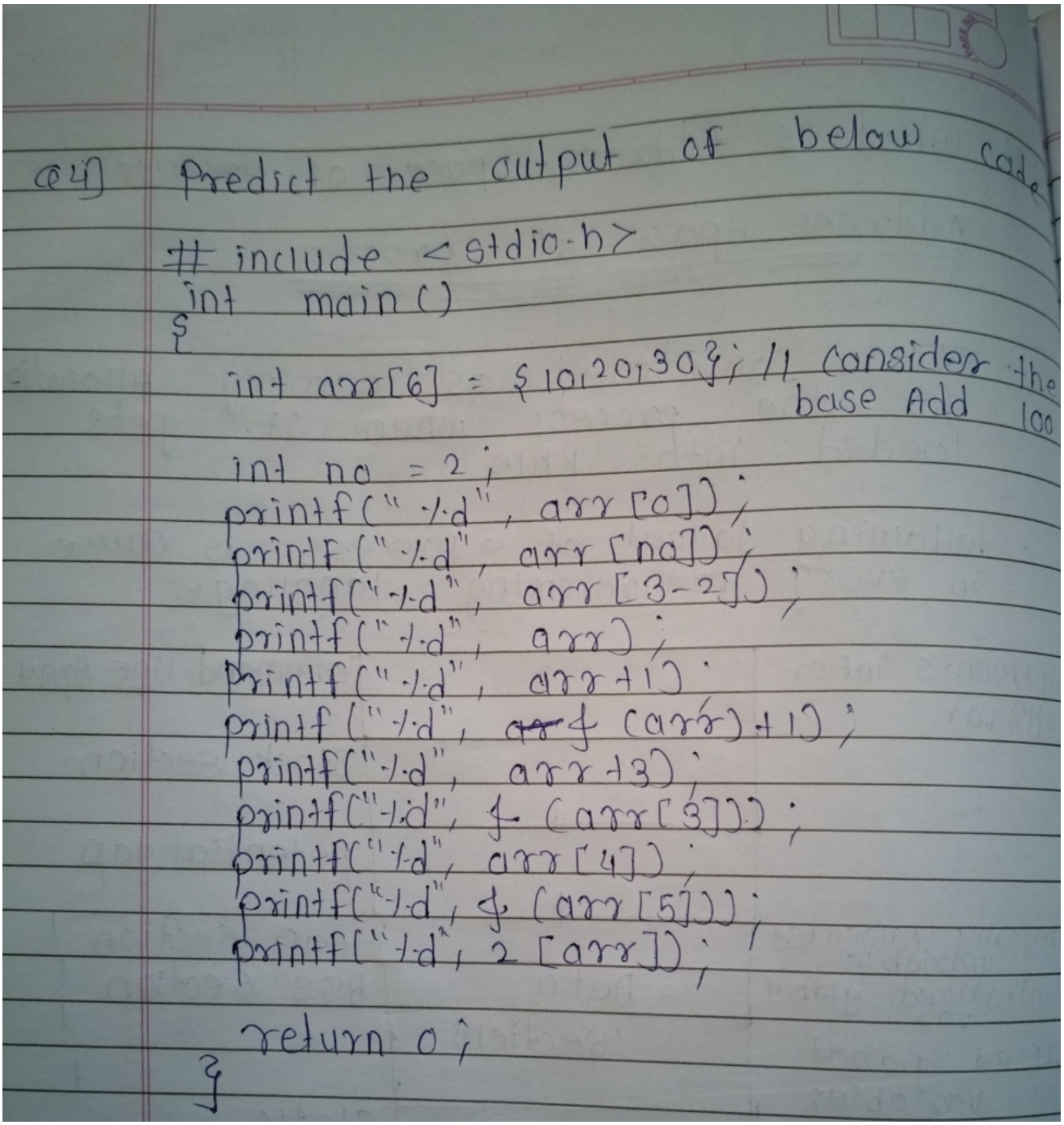
→ Address space of process

- It is considered as memory Allocation for the process when it gets loaded into RAM.

- following layout of a process is same in every programming language



- consider above section of the executable file.



Q4] Predict the output of below code.

```
#include <stdio.h>
int main()
{
    int arr[6] = {10,20,30}; // consider the base Add 100

    int no = 2;
    printf("%d", arr[0]);
    printf("%d", arr[no]);
    printf("%d", arr[3-2]);
    printf("%d", arr);
    printf("%d", arr+1);
    printf("%d", *(arr)+1);
    printf("%d", arr+3);
    printf("%d", *(arr[3]));
    printf("%d", arr[4]);
    printf("%d", *(arr[5]));
    printf("%d", 2[arr]);

    return 0;
}
```

output:-

int no=2

	0	1	2	3	4	5
arr	10	20	30	0	0	0
	100	104	108	112	116	120	124

arr[0] = 10
arr[no]) = 30
arr[3-2]) = 0 - 30 = -30
arr = 100
arr+1 = 104
*(arr)+1 = 124
arr+3 = 112
*(arr[3])) = 112
arr[4] = ~~116~~ 0
*(arr[5])) = 120
2[arr] = 30

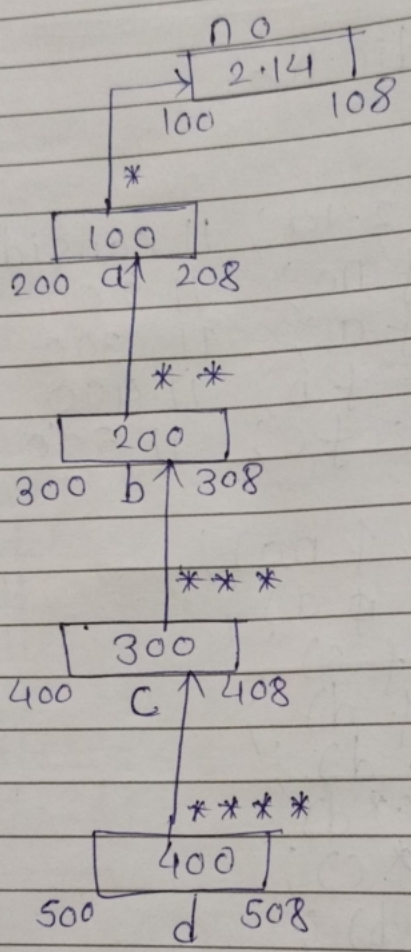
Q5] Predict the output of below code & draw its diagrammatic representation.

```
#include <stdio.h>
int main()
{
double no = 2.14; // consider add 10
double *a = &no; // 200
double **b = &a; // 300
double ***c = &b; // 400
double ****d = &c; // 500

printf("%d", &no);
printf("%d", &a);
printf("%d", &c);
printf("%d", &d);
printf("%d", *d);
printf("%d", **d);
printf("%d", **c);
printf("%d", *b);

return 0;
}
```

O/P:-

```
&no = 100
 a  = 100
&c  = 400
&d  = 500
 *d = 400
**d = 200
**c = 100
 *b = 200
```

memory representation

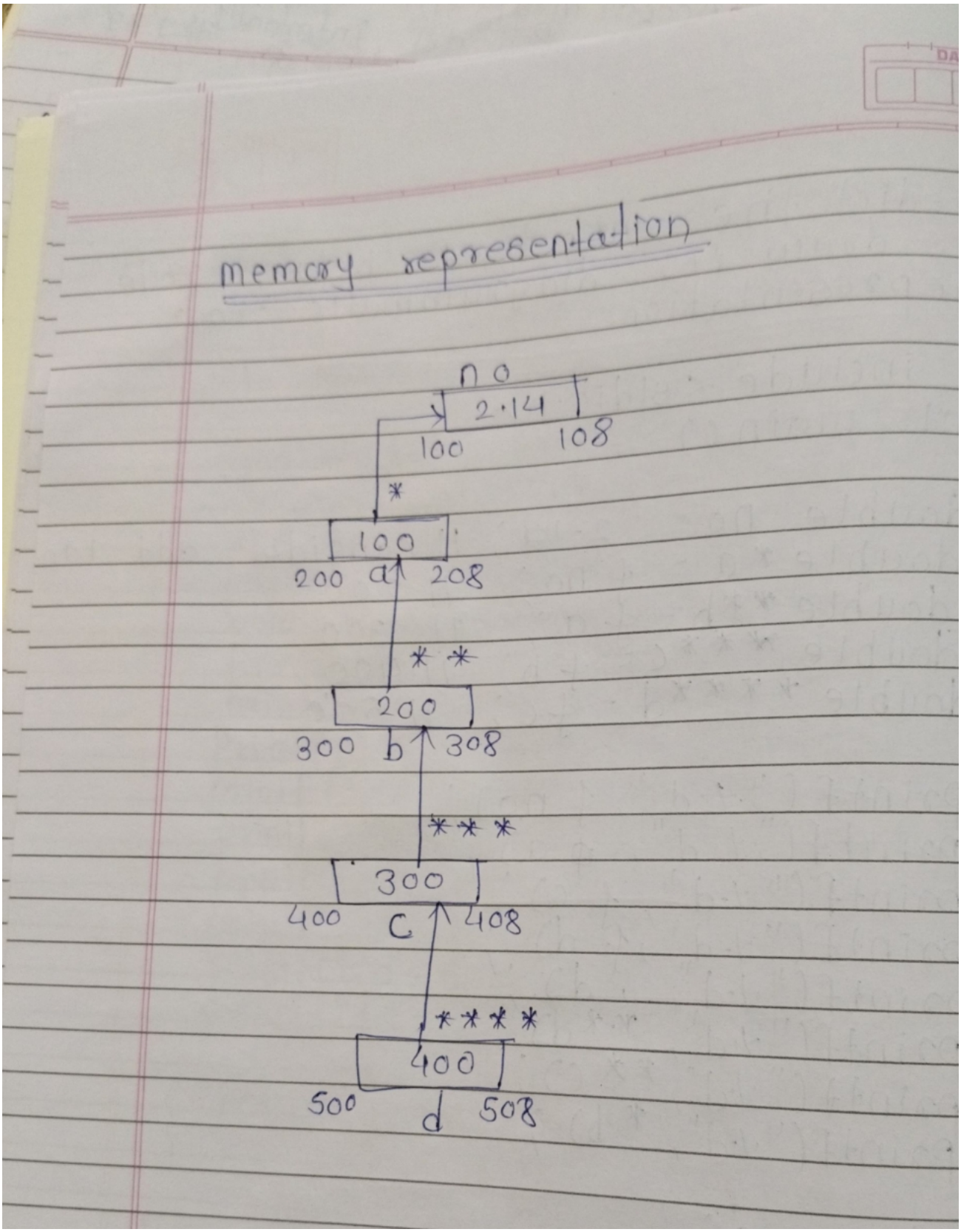


06] Predict the output of below code & draw its diagrammatic representation

```
#include <stdio.h>
int main()
{
    double no = 2.14; // consider Add is 100
    double *a = &no; // 200
    double **b = &a; // 300
    double ***c = &b; // 400
    double ****d = &c; // 500

    printf("%d", sizeof(no));
    printf("%d", sizeof(a));
    printf("%d", sizeof(b));
    printf("%d", sizeof(c));
    printf("%d", sizeof(d));
    printf("%d", sizeof(**d));
    printf("%d", sizeof(**c));
    printf("%d", sizeof(****d));
    printf("%d", sizeof(***c));

    return 0;
}
```

o/p:

```
8   // datatype of var no has double.
8   // a is the pointer which points to no.
8   // b is pointer which points to a.
8   // c is pointer which points to b & store Address
8
8
8
8
```


memory representation
no
2.14
100
108
*
100
200
a
208
**
200
300
b
308
***
300
400
C
408
****
400
500
d
508

Q7) What is pointer in c programming? why to use it.

→ Pointer:-

- pointer is considered as derived datatype in C & C++
- pointer is considered as variable which store the memory Address
- pointer is such a special variable which can store the memory Address of anything the power of pointer is to fetch the data whose address in it.

for example:-

```
int *ip = &i;
```

- ip is a pointer which points to integer datatype currently it holds the value of i
- the size of every pointer is 8 byte because it holds the memory address is of type unsigned long.
- every pointer data fetching capacity is based or depend on datatype

- * operater is used with pointer to fetch the data that pointed variable.

- * is called as dereference operator.

Q8) What are different usages of pointer explain with example.

→ 1) We can create pointer of Array

- we can create the pointer of Array

e.g:-

```
# include <stdio.h>
int main ()
{
  int Arr[5] = { 10, 20, 30, 40, 50 };

  int *p = Arr;
  int *q = &(Arr[4]);

  return 0;
}
```

here we create the Array Arr having 5 elements in has ~~de~~ integer datatype

- p is pointer which points to
- q is pointer which points to 4th elements of Arr & store the Address.

2] Pointer Arithmatic

- In case of a pointer this arithmatic operations are not like normal operator that perform on numeric values.
- pointer Arithmatic support operation like increment & decrement .

for example:

```
# include <stdio.h>
 int main ()
{
 ~~int arr [2] = { 1, 2 };~~
  int  a = 10;
  int * ptr = &a;

  printf (" ptr = %u", ptr);
  ptr ++;
  printf (" ptr = %u", ptr);

 return 0;
  }
```

3] Dynamic memory Allocation

1) Using malloc () function by using pointer

```
int * ptr = (int*) malloc (5* sizeof (int));
```

2) Calloc ()

- in case of calloc () we pass 2 parameter Arguments.
- first is no-of-elements, size-of-element.

e.g:-

```
ptr = (float*) calloc (25, sizeof (float));
```

3) Realloc ()

- used for resize the Allocated memory.

e.g:-

```
int * ptr = (int*) malloc (5 * sizeof(int));
```

4) Pointer to pointer.

- One pointer ~~store~~ points to another pointer

e.g:-

```
#include <stdio.h>
int main ()
{
    int var=10;
    int * ptr1 = &var;
    int **ptr2 = &ptr1;

    printf("Var: %d\n", var);
    printf("ptr1: %d\n", *ptr1);
    printf("ptr2: %d\n", **ptr2);

    return 0;
}
```

eg) write a program to demonstrate the different types of pointer which are pointing to primitive data types?

→

1) for integer datatype

```
#include <stdio.h>
int main()
{
    int a = 10;
    int *p = &a;

    printf("%d", &*p);

    return 0;
}
```

2) for character datatype

```
#include <stdio.h>
int main()
{
    char x = 'p';

    char *ptr = &x;

    printf("%c\n", *ptr);

    return 0;
}
```

3) for float datatype

```
# include <stdio.h>
int main ()
{
  float no = 33.14;
  float *ptr = &no;

  printf("%f", *ptr);

  return 0;
}
```

4) for double datatype

```
# include <stdio.h>
int main ()
{
  double d = 40.50;
  double *p = &d;

  printf ("%d", *p);

  return 0;
}
```

Q 10] read the below statements & write reading statements for the same.

int no = 10;

- no is variable having integer datatype initalizing with value 10.

a] int *p = &no;

- p is a pointer which points to no having integer datatype it holds the address of no & fetch the value in it.

b] char ch = 'A';

- ch is a variable has ~~dat~~ datatype is char & initalizing with value A.

char *chptr = &ch;

- chptr is pointer which points to ch & currently holds the address of ch having character datatype & fetch the value in it.

d] double d = 10202020;
double *dbptr = &d;

- d is variable having double datatype initalizing with value 10202020.

- dbptr is pointer which points to d & currently holds address of d variable having double datatype & fetching the value in it.