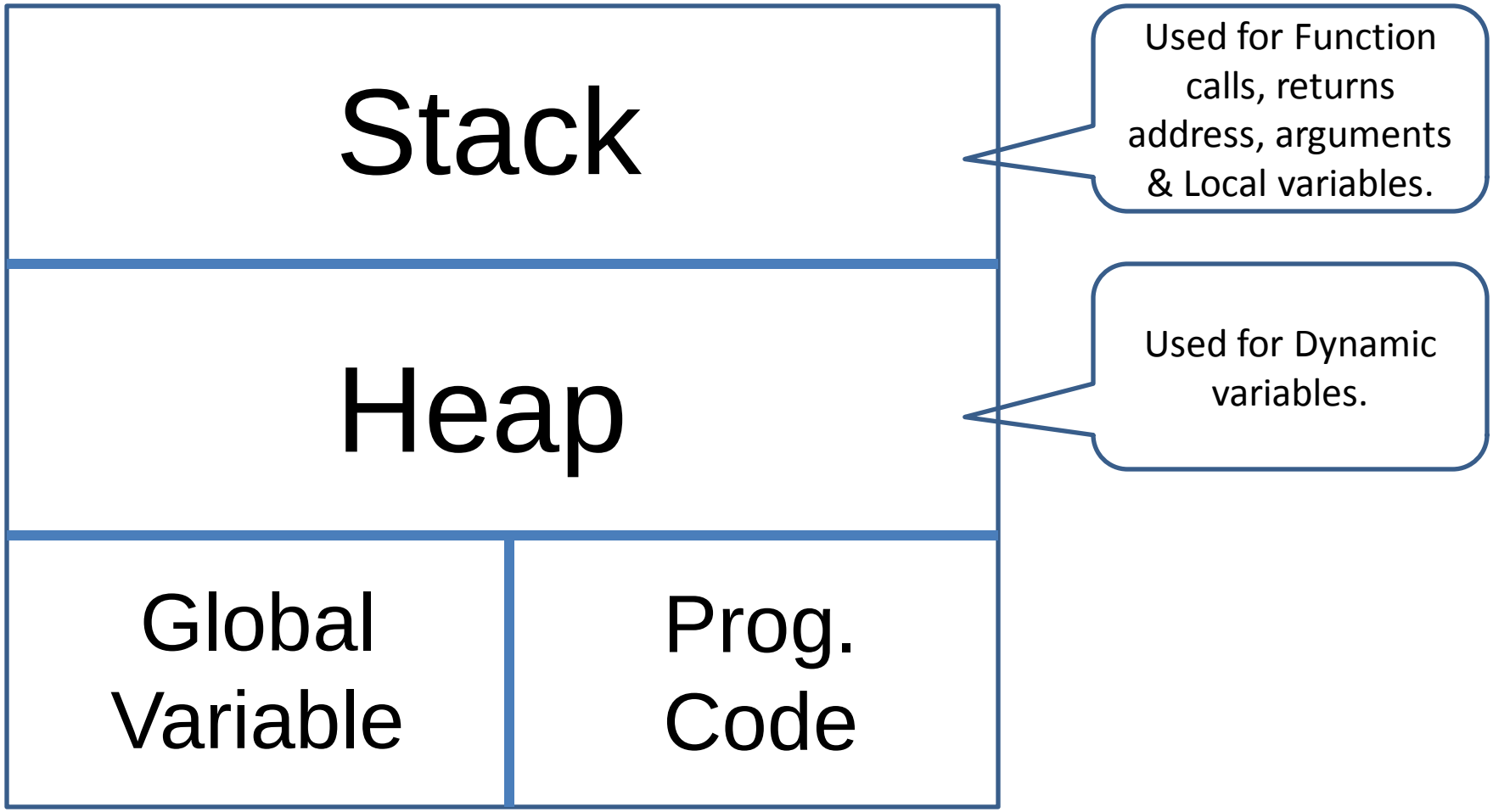
POINTERS

CLASS XII

C++ Memory Map


Stack
Heap
Global Variable
Prog. Code
Used for Function calls, returns address, arguments & Local variables.
Used for Dynamic variables.

Declaration & Initialization of Pointers

Datatype * var_Name

eg. int * iptr

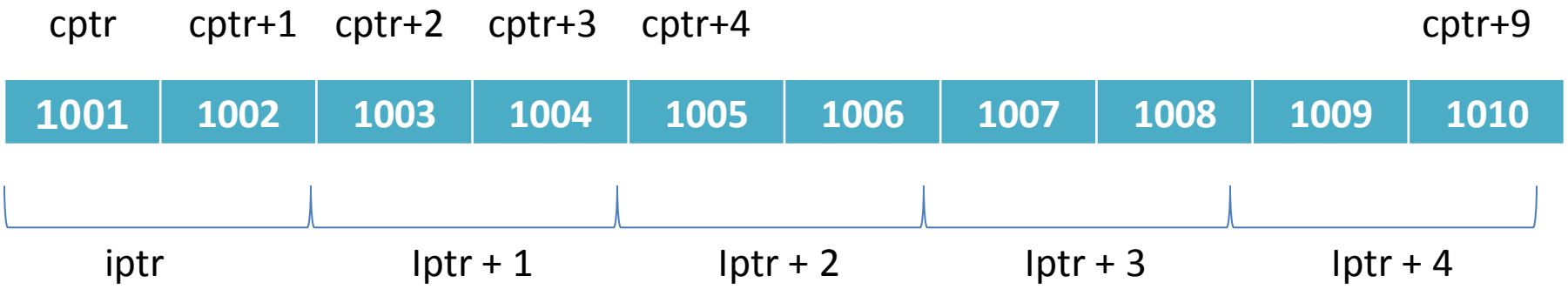
char * cptr

Initialization :

int x = 25; char c = ‘A’

iptr = &x; cptr = &c ;

Pointer Arithematic




```
iptr = &Ar ;        // i. e.  1001

iptr++              // iptr   become  1003.


Cptr = &Ar ;        // i. e.  1001

cptr++              // iptr  become   1002.
```

Dynamic Memory Operation

- Use of ‘ new’ and ‘ delete’ keyword

eg.

```
int *iptr ;
iptr = new int ;
Iptr = 25;
```

OR

```
int *iptr  = new int (25) ;
```

```
char *cptr ;
cptr  = new char ;
cptr  = ‘A’ ;
```

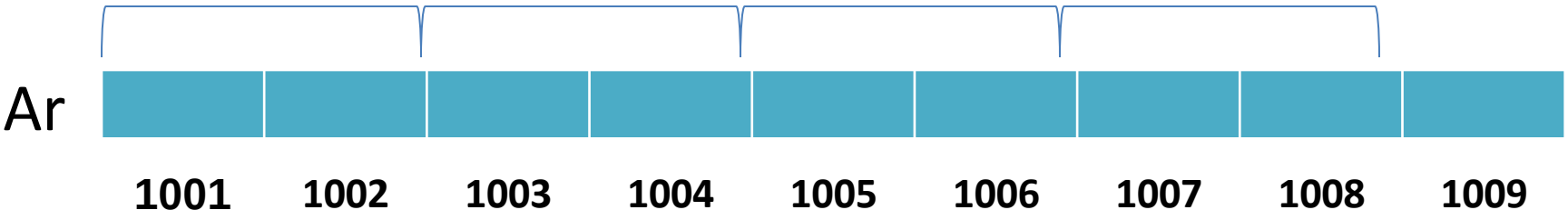
```
char *cptr  = new char (‘A’) ;
```

```
float *fptr ;
fptr  = new float;
fptr = 15.25 ;
```

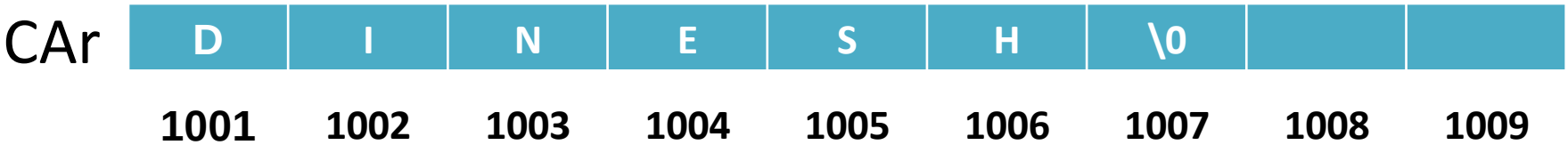
```
float *fptr   = new float(15.25) ;
```

Creation of Dynamic Array

- int *Ar = new int [10] ;



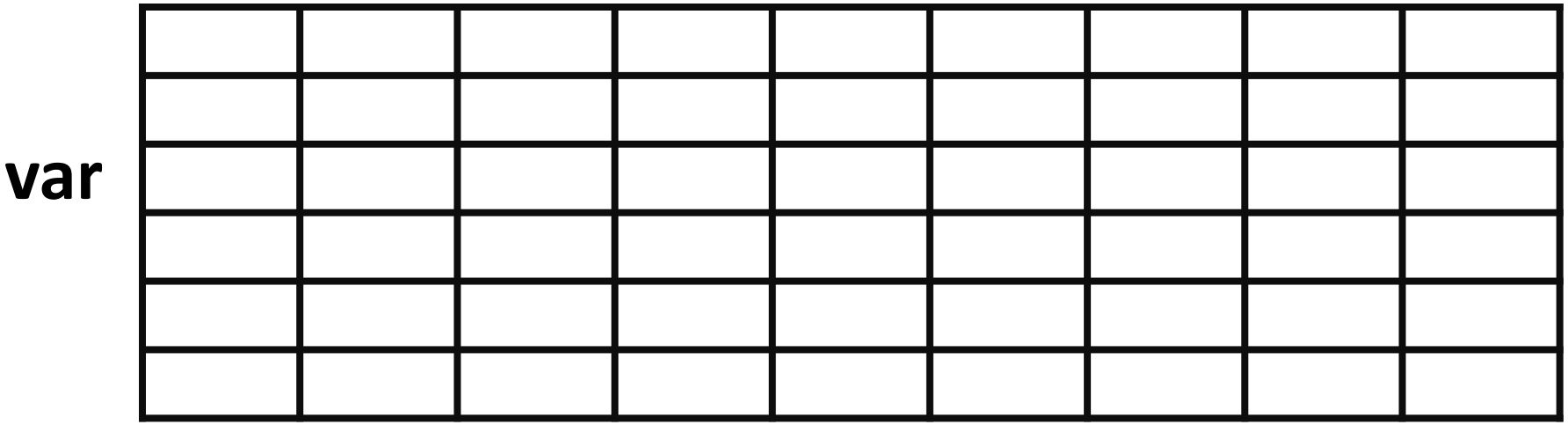
- char *CAr = new char [10] ;



Creation of 2-D Array

- int * var = new int [row * col];

eg. int * var = new int [6 * 9] ;



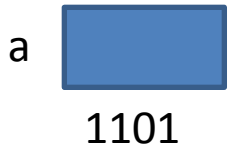
Deletion of Array

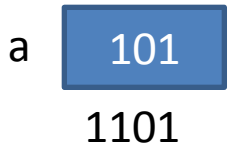
- delete iptr; //Pointer deletion
- delete [10] iptr; // 1-D pointer deletion

Memory Leak

Pointers and Array

```
void main()
{ int age[5]={10,20,30,40,50};
  int *a;
  a=age;
  cout<<"*a = "<<*a<< " @ location  a = " << a;
  cout<<endl;
  cout<<"*age = "<<*age <<" @ location age = "<<age ;
}
```

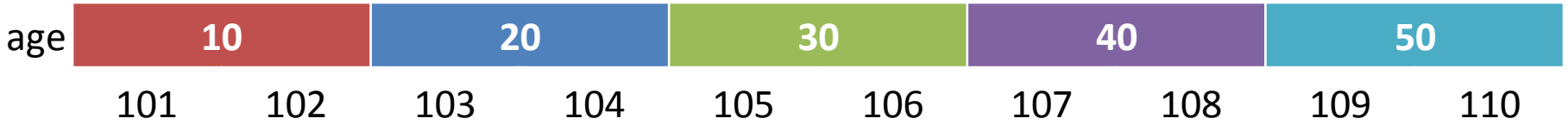


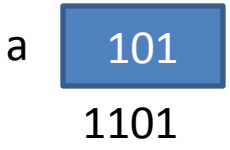


OUTPUT :

**a = 10 @ location a = 101*

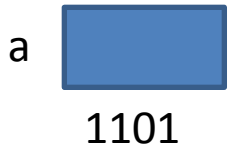
**age = 10 @ location age = 101*

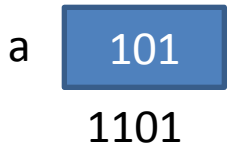




also

```
void main()
{ int age[5]={10,20,30,40,50};
  int *a;
  a=age;
  cout<<"*a = "<<*a +1<< " @ location  a = " << a+1;
  cout<<endl;
  cout<<"*age = "<<*age +1<<" @ location age = "<<age+1 ;
}
```

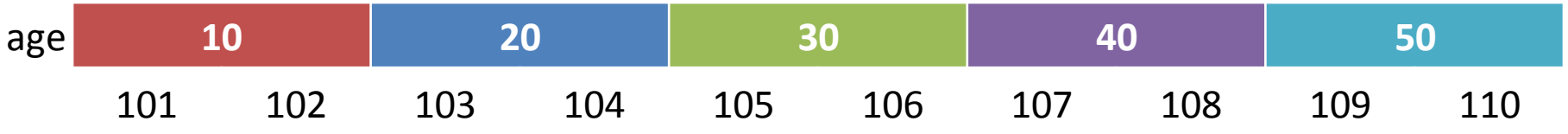


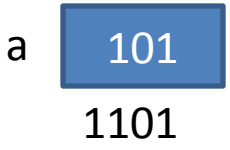


OUTPUT :

*a = 20 @ location a = 103

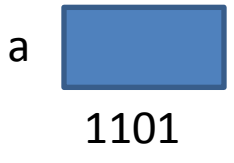
*age = 20 @ location age = 103

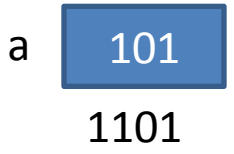




similarly.....

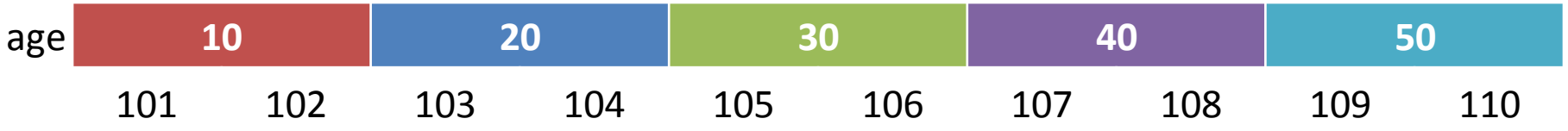
```
void main()
{ int age[5]={10,20,30,40,50};
  int *a;
  a=age;
  cout<<"*a = "<<*a<< " @ location  a = " << ++a;
}
```

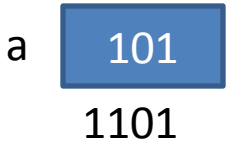




OUTPUT :

*a = 20 @ location a = 103



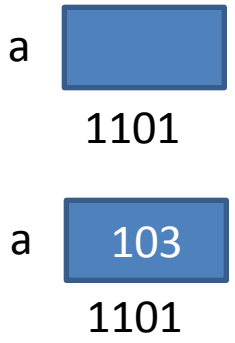


But……

```
void main()
{ int age[5]={10,20,30,40,50};
  int *a;
  a=age;
  cout<<"*a = "<<*++a<< " @ location  a = " << ++a;

  age=a;                                          //error
  cout<<"*age = "<<*++age<< " @ location  a = " << ++age;  //error

}
```



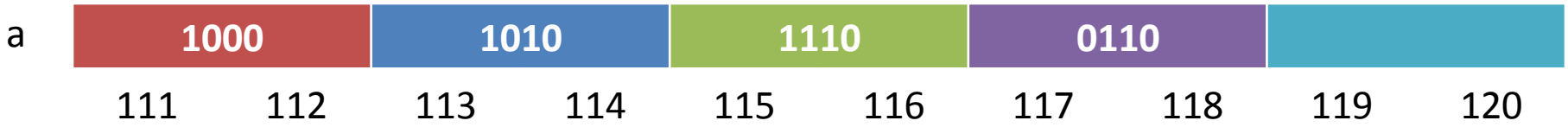
Not allowed because ‘age’ is defined as array not as pointer

Array of pointer

int va=10, vb=20, vc=30, vd=40, ve=50;

int *a[5];

a[0]=&va, a[1]=&vb, a[2]=&vc, a[3]=&vd;



cout<< a ;	**//address of pointer(0)**	**111**
cout<< *a;	**//address stored in pointer(0)**	**1000**
cout<< **a;	**//value of address of pointer(0)**	**10**
cout<< *(a+1);	**//address stored in pointer(1)**	**1010**
cout<< ++*a;	**//next address of address stored in pointer(1)**	**i.e 1012**

(which is not defined)

Pointers with 2D arrays

```
int x[3][5]={{1,2,3,4,5}, {6,7,8,9,10},{11,12,13,14,15}};
int *a=&x[0][0];
```

x or *x or a	=	&x[0][0]	
**x	=	1	
*x+1	=	&x[0][1]	
*(x+1)	=	&x[1][0]	
*a	=	x[0][0]	= 1
*a+50	=	x[0][0]+50	= 51
*(a+8)	=	x[1][3]	= 9

X[3][5]

	0	1	2	3	4
0	1	2	3	4	5
1	6	7	8	9	10
2	11	12	13	14	15

*a	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	111	113	115	117	119	121	123	125	127	129	131	133	135	137	139

```
#include<iostream.h>
void main()
{  int x[3][5]={{1,2,3,4,5}, {6,7,8,9,10},{11,12,13,14,15}};
   int *a=&x[0][0];
   cout<<"x = "<<x<<",\t  *x = "<<*x<<",\ta = "<<a<<",\t&x[0][0] = "<<&x[0][0] <<"\n\n";
   cout<<"**x = "<<**x <<"\n\n";
   cout<<"*x+1 = "<<*x+1 <<",\t  &x[0][1] = "<< &x[0][1]<<"\n\n";
   cout<<"*(x+1) = "<<*(x+1) <<",\t  &x[1][0] = "<< &x[1][0]<<"\n\n";
   cout<<"*a = "<<*a <<",       x[0][0] = "<<x[0][0]  <<"          i.e. = 1"<<"\n\n";
   cout<<"*a+50 = "<<*a+50<<",    x[0][0]+50 = "<< x[0][0]+50 <<"       i.e. = 51"<<"\n\n";
   cout<<"*(a+8) = "<< *(a+8) <<",    x[1][3] = "<<x[1][3] <<"          i.e. = 9"<<"\n\n";
}
```

String pointer array i.e. 2D array

```
char *day[7] ={    “Sunday”,
                   “Monday”,
                   “Tuesday”,
                   “Thursday”,
                   “Friday”,
                   “Saturday”
              };
```

Means :

```
cout<<day[0];                                   //Sunday
```

Pointer and Constant

- int n=50;
- int *ptr = &n;
- ++(*ptr); //ok i.e. 51

- int * const p; //ok (cont pointer assign)
- ++(*p); //ok i.e. 51
- ++p; //wrong p is const pointer

- const int *pt; //ok (pointer to hold const value)
- ++(*pt) //wrong (value can't increase)
- ++pt //ok (pointer can be increase)

- const int * const cptr; //const. pointer to hold const value
- ++(*cptr) //not allowed (value is const.)
- ++cptr //wrong (pointer is const.)

Thanks……..

By : DINESH PATEL
PGT [CS]
KV KARANJA