Assignment - 19

1. what is difference in malloc and new, free and delete.

-> ◦ malloc () is a function used in c as well as in c++ for the purpose of dynamic memory allocation.

◦ new is a operator used in c++ for the purpose of dynamic memory allocation. Java

◦ new is a operator and malloc () is the standard library function. (stdlib).

◦ New : Memory initialized by the new operator is allocated in a heap.

◦ malloc () : Memory initialized by malloc () is allocated on a heap.

◦ New : It returns starting address of a memory.
malloc : It also returns the address of a memory

◦ New : Syntax : type variable = new type (parameter_list);

int A[2] ; int *p = new int [2]

◦ Malloc () : Syntax: type variable_name = (type*) (malloc (sizeof (type)

free() & delete

- free : In c/c++ programming to deallocate the memory free() function is used.

- delete : In c++ programming to deallocate memory delete operator is used.

- free : Syntax -

```
free (variable);
```

- delete : Syntax -

```
delete variable [parameter];
```

- malloc & new

- new operator constructs object i.e it calls the constructor to initialize an object while malloc() does not call constructor.

- new operator invokes constructor & delete operator invokes destructor.

2. write a Syntax which is used to allocate memory for 10 integers dynamically - using malloc.

->

```
# include <stdio.h> <iostream>
# include <stdlib.h>
using namespace std;

int main()
{
    int Arr[10]; // int size = 10;
    int *ptr = (int*) malloc (10 * sizeof(int));

    free (ptr);

    return 0;
}
```

Sudo code

```
int size = 10;
int * ptr = NULL;
ptr = (int *) malloc (size * sizeof(int));
```

- User Input - Accept [10] integers dynamically allocate memory for those.

```
# include <iostream>
# include <stdlib.h>
  using namespace std;

int main()
{
   int ilength=0;
   cout<<" Enter Length of Integers : "<<endl;
   cin >> ilength;

   int *ptr;
   ptr = (int *) malloc ( ilength * sizeof(int));
                                    // Dynamically memory
                                    // allocation
   for (int i=0 ; i <ilength ; i++)
   {      cout << " Enter Number:" << endl;
          cin >>  * (ptr +i) << endl ;
   }
   cout << " Your Integer elements are :" << endl;

    for ( int i=0 ; i < ilength ; i++ )
    {
        cout << * (ptr+i) << endl ;
    }
    free (ptr);
    return 0;
}
```

3. write syntax which is used to allocate memory for 10 integers dynamically using realloc.

→

```
# include <stdlib.h>
# include <iostream>
using namespace std;

int main()
{
    int *ptr = NULL;

    ptr = (int *) realloc (ptr, 10* sizeof(int));

    free (ptr);
    return 0;
}
```

4. what is mean by dangling pointers

5. What is the return value of malloc() function if memory manager is unable to allocate the memory.

—>

- If memory manager is unable to allocate the memory then the return value of malloc function is NULL.
- In this way malloc indicates that the memory allocation request is failed.

6. Write a syntax which is used to allocate memory for N floats dynamically using malloc. Accept the value of N from user at run time.

```
#include <stdlib.h>
#include <iostream>
using namespace std;

int main()
{
    float fNum = 0.0f;
    cout << "Enter Number : " << endl;
    cin >> fNum ;

    float * ptr;
    ptr = (float *) malloc ( fNum * sizeof (float));
```

```
for (int i = 1  ; i <= fNum ; i++){

        cout << "Enter Numbers :" << endl ;
        cin >> *(ptr+i)
    }
    cout << "You Entered Numbers :" << endl ;
    for ( int i = 1  ; i <= fNum ,i++)
    {
        cout << *ptr +i ;
    }
    free (ptr);
    return 0 ;
}
```

7. Allocate dynamic memory for array of 5 elements where each element is below structure.

```
struct hello
{
    float f ;
    int i ;
};
```

->

```
# include < stdio.h>
# include < stdlib.h>

struct hello
{
    float f;
    int i ;
};
```

```
int main()
{
    struct hello *ArrayPtr;

    ArrayPtr = (struct hello *) malloc (5 * sizeof (struct hello));

    free (ArrayPtr);

    return 0;
}
```

8. Explain internal working of new operator in detail?

—> ◦ new operator is used to allocate memory Dynamically.

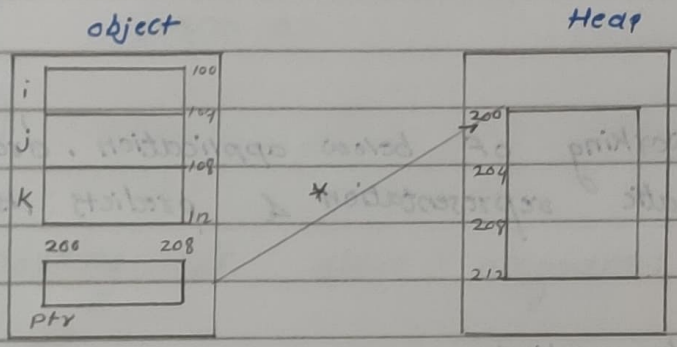
◦ new operator internally call malloc function.

◦ but, in case of new there is no need of typecasting.

◦ For example:

```
class Demo {
    public:
        int i,j,k;
        Demo() { };          // Default constructor
        ~Demo() { };         // Destructor
    }; Display();
```

```
int main()
{
    Demo *ptr = new Demo;   // Auto constructor &
    ptr -> Display();       //      destructor call
    delete ptr;
    return 0;
}
```

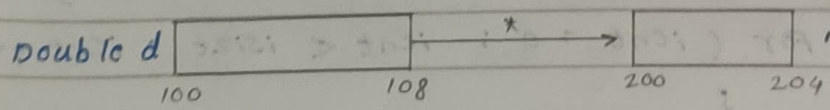


output:
// constructor
// Destructor
// Display

- in above example we have create object using new keyword dynamically, memory is allocated inside a heap.
- new and delete will create and destroy object
- constructor and destructor automatically gets called as soon as object gets created.

9. why concept of Down casting is not allowed?

→
- if a pointer points to a data have a less size as compare to pointers capacity, then it is called a Down casting.



- The concept of down casting is not allowed because it can lead to runtime error.

10. Explain working of below application, draw its diagrammatic representation & predicts its output.

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int isize = 0;
    int *p = NULL;
    int icnt = 0;

    printf("Enter number of element : \n");
    scanf("%d", &isize);

    p = (int *) malloc (isize * sizeof (int));

    printf ("Please Enter elements : \n");
    for (icnt = 0 ; icnt < isize ; icnt++)
    {
        scanf ("%d", &p[icnt]);
    }
```

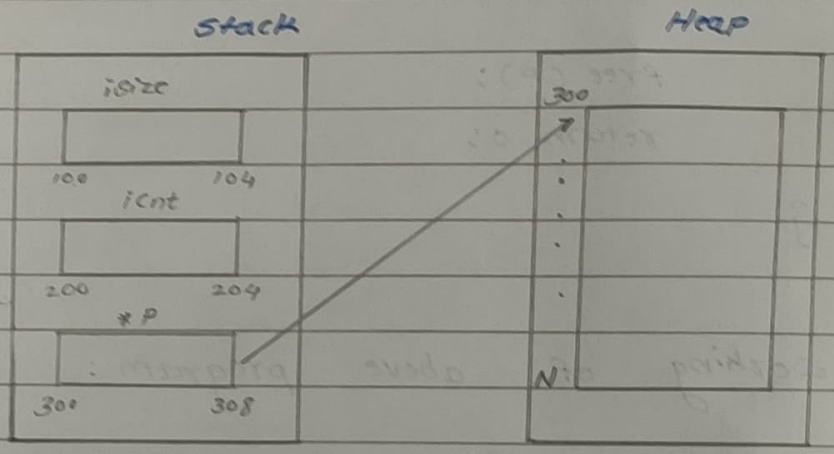
```
    printf ( " Entered Elements : \n");
    for ( icnt = 0 ; icnt < size ; icnt++)
      {
          printf ( " %d ", p(icnt) ;
      }

      free (p);
      return 0;

  }
```

-> o working of above program :

- In above coode, one integer pointer (*p) and two integer variables are declared as isize & icnt (icnt is a counter).
- isize accept number of element from user, and store.
- *p - p is used to initialize / allocate memory dynamically using malloc function.
- malloc accepts isize * sizeof (int) paramter & allocate memory accordingly on heap.
- p is initialize with elements which user gives.
- For loop is used to accept element from user one by one.
- elements are stored from 0 index to isize. at a location of p &P[icnt].

- Another for loop is used to display Entered elements.

- Diagramatic Representation:



- output : Enter no. of element : 3
Please Enter element : 2
4
6
Entered Elements are : 2
4
6