Answer Script

Question No. 1-a =
Explain Stack and Heap memory.
Answer No. 1-a
Stack Memory : Stack Memory is a kind of memory that is used to store local variables , functions , temporary data and its size is typically fixed. As it's fixed size, it's faster and efficient. **Heap Memory :** Heap memory is a memory that is used for dynamic memory allocation. Its size is not fixed , we programmers can increase or decrease the size of it. It is generally complex and slower compare to stack memory

Question No. 1-b
Why do we need dynamic memory allocation? Explain with examples.
Answer No. 1-b
Stack memory is fixed , it's not possible to change the size of it at runtime, but sometimes we need to add or delete some data from our storage or memory , Stack memory dont allow to do that, but heap memory allows it, Thats mean dynamic memory allocation allows us to allocate memory for any objects at runtime. **Example** : suppose we have an array called **arr[5]** we can initialize this by **int arr[5] ={1,2,3,4,5}** in normal array we cannot increase the value of it, if we need to increase we have to create another array(**arr2[7]**) bigger size then of previous one and copy from first one to second array and add more 2 or 3 values we want to add. But the problem is the first array is not delete from the memory but we don't need this array.

But in dynamic memory allocation we can delete that first one. To do this we have to declare the array in a dynamic way.

```
int *a = new int[5];
int *b = new int[7];
delete[] a;
```

Now the first array a will be deleted and only b array will remain.

Code :
1.Normal array :

```cpp
#include<bits/stdc++.h>
using namespace std;
int main(){

    int arr[5];
    for(int i=0; i<5; i++){
        cin >> arr[i];
    }
    int b[7];
    for(int i=0; i<5; i++){
        b[i] = arr[i];
    }
    b[5] = 60;
    b[6] = 70;
    for(int i=0; i<7; i++){
        cout << b[i] << " ";
    }
    return 0;
}
```

Dynamic array :

```
#include<bits/stdc++.h>
using namespace std;
int main(){

    int * arr = new int[5];
    for(int i=0; i<5; i++){
        cin >> arr[i];
    }
    int *b = new int[7];
    for(int i=0; i<5; i++){
        b[i] = arr[i];
    }
    b[5] = 60;
    b[6] = 70;
    for(int i=0; i<7; i++){
        cout << b[i] << " ";
    }

    delete[] arr;
    return 0;
}
```

Question No. 1-c
How to create a dynamic array? What are the benefits of it?
Answer No. 1-c
We can create a dynamic array using the **new operator** . Other things are like normal arrays . Create dynamic array : **Data type * name_array = new data_Type[size];** Ex : **Int * arr = new int[5];**

Benefits :
Dynamic array allows to increase or decrease or even delete any any from the memory , the size of the array is determined at runtime so we can easy increase or decrease the size of the array.Its efficient , only consume that we need only , others things can be remove from memory , this is the main benefits .

Full code to Create a dynamic array :

```
#include<bits/stdc++.h>
using namespace std;
int main(){

    int * arr = new int[5];
    for(int i=0; i<5; i++){
        cin >> arr[i];
    }
    int *b = new int[7];
    for(int i=0; i<5; i++){
        b[i] = arr[i];
    }
    b[5] = 60;
    b[6] = 70;
    for(int i=0; i<7; i++){
        cout << b[i] << " ";
    }

    delete[] arr;
    return 0;
}
```

Question No. 2-a

How does class and object work? How to declare an object?

Answer No. 2-a

Class is a blueprint or template of different kind of objects , in real life example we can consider a car as a class , so a car has a color , pice, speed etc, those color , price , speeds are object of car.

Class ClassName{};
Ex : class Car{};

After creating the class then we can create objects of this class .create object :

className objectName;
Ex : Car tata;

Now **tata** is the object of class **car .** now we can define values fo the objects .

tata.name = “tata”;
tata.speed = 232;
tata.color = “Black”;

Example code :

```
#include<bits/stdc++.h>
using namespace std;
class student{
    public:
    int id;
    int roll;
};
int main(){
    student s;
    s.id = 1;
    s.roll = 45;
    cout << s.id <<  " " << s.roll << endl;
}
```

Question No. 2-b

What is a constructor and why do we need this? How to create a constructor show with an example.

Answer No. 2-b

Constructor is a special kind of method or function in a class where we can initialize the objects of that class . we can use multiple constructor for a single class. We don't need to declare a constructor via an object , constructor is automatically called. We should use constructor to avoid writing multiple object definitions multiple times. Example :

Object 1:

```
Car tata; //declare object for class car
tata.name ;
tata.speed;
tata.color;
```

Object 2:

```
Car  bata; //declare object for class car
bata.name ;
bata.speed;
bata.color;
```

So if we have 100 objects we have to declare them 100 times and initialize those values . so it's a hard one. It's difficult so we need a constructor to solve this problem.

By using constructor:

```
Car tata("Tata",230,"Black");
Car bata("Bata",290,"white");
```

This is short an useful and also user friendly

Example :

```cpp
#include <bits/stdc++.h>
using namespace std;
class Student{
    public:
    int id;
    int roll;
    char section;
    char name[100];
    Student(int i,int r,char s,char* n){
        id = i;
        roll = r;
        section = s;
        strcpy(name,n);
    }
};
int main(){
    char name1[100] = "Nayeem";
    char name2[100] = "Fahim";
    Student nayeem(1,45,'B',name1);
    Student fahim(2,34,'B',name2);
    cout << nayeem.id << endl;
    cout << nayeem.roll << endl;
    cout << nayeem.section << endl;
    cout << nayeem.name << endl;
    cout << fahim.name << endl;
    cout << fahim.id << endl;
    cout << fahim.roll << endl;
    return 0;
}
```

Question No. 2-c
Create a class named Person where the class will have properties name(string), height(float) and age(int). Make a constructor and create a dynamic object of that class and finally pass proper values using the constructor.
Answer No. 2-c

```cpp
#include <bits/stdc++.h>
using namespace std;
class Person{
    public:
    char name[100];
    float height;
    int age;

    Person(char* n,float h, int a){
        strcpy(name,n);
        height = h;
        age = a;
    }
};
int main(){
    char name1[100] = "Nayeem";
    Person * nayeem = new Person(name1, 100.1,23);
    cout << nayeem->name << endl;
    cout << nayeem->height << endl;
    cout << nayeem->age << endl;

    return 0;
}
```

Question No. 3-a

What is the size that an object allocates to the memory?

Answer No. 3-a

Object itself does not have any size i mean it does not consume any memory , but the size of the object totally depends on the object's inside data type. I will explain a little bit. For example **tata** is an object of **car** class , so tata does not contain any space itself. But tata object's has some data type value, like it hase int model no, char model name, int speed, float miles etc. so those int,float,char consume memory , we know that an integer member typically occupies 4 bytes, while a double member occupies 8 bytes.char contain 1 byte. So the total size of an object will be the **sum of the total data type.**

Question No. 3-b

Can you return a static object from a function? If yes, show with an example.

Answer No. 3-b

Yes , It's Possible . Here we create a function to return my **toyota** object, but the datatype will be a custom data type.

```
#include <bits/stdc++.h>
using namespace std;
class car{
    public:
    int seiral;
    int model;
    char section;
    char name[100];

    car(int i,int r,char s,char* n){
        seiral = i;
        model = r;
        section = s;
        strcpy(name,n);
```

```
    }
};

car fun(){
    char name[100] = "Toyta";
    car toyta(1,23,'A',name);
    return toyta;

}
int main(){
    car toyta = fun();
    cout << toyta.seiral << endl;
    cout << toyta.model << endl;
    cout << toyta.section << endl;
    cout << toyta.name << endl;
    return 0;
}
```

Question No. 3-c
Why do we need -> (arrow sign)?
Answer No. 3-c
We need a “->” sign to see or access the data or value of the dynamic object . Normal objects work with Dot(.) but dynamic objects cannot read this, we need an arrow sign to see the data of any dynamic object.

Question No. 3-d
Create two objects of the Person class from question 2-c and initialize them with proper value. Now compare whose age is greater, and print his/her name.

Answer No. 3-d

```cpp
#include <bits/stdc++.h>
using namespace std;
class Person{
    public:
    char name[100];
    float height;
    int age;

    Person(char* n,float h, int a){
        strcpy(name,n);
        height = h;
        age = a;
    }
};
int main(){
    char name1[100] = "Nayeem";
    char name2[100] = "Fahim";
    Person * nayeem = new Person(name1, 100.1,13);
    Person * fahim = new Person(name2, 56.4,23);

    int age1 = nayeem->age;
    int age2 = fahim->age;
    if( age1 > age2 ){
        cout << nayeem->name <<" : " <<  age1 <<  endl;
    }else if (age1 < age2)
    {
         cout << fahim->name <<" : " <<  age1 <<  endl;
    }else cout << "Both are Equeal" << endl;

    return 0;
}
```