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 contains all the information of a function until it is completed, after completing the function all the information goes deleted automatically from memory.
It does not provide memory allocation/allocation facilities after compile time which happens automatically.

Heap Memory: Heap Memory is a kind of memory that contains all the information of a function even after completing it, after completing the function all the information remains in memory if you wish you can delete them or not.
It provides memory allocation/deallocation facilities even after compile time which happens both automatically or manually.

Question No. 1-b

Why do we need dynamic memory allocation? Explain with examples.

Answer No. 1-b

We need Dynamic memory allocation to acquire some extra facilities rather than Static memory allocation like - memory allocation/deallocation manually at run time, remaining information of a function even after completing it etc.

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    string *str = new string;
    *str = "dynamically ekta string allocate korlam";
    cout << *str << "\n";
    cout << "string er kaj shes ekhon deallocate korbo\n";
    delete str;  //this deallocation does not allow by static memory
    return 0;
}
```

Question No. 1-c

How to create a dynamic array? What are the benefits of it?

Answer No. 1-c

Dynamic array creation:

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    int *arr = new int[10];      //create a dynamic array
    for(int i = 0; i < 10; i++) arr[i] = i+1;

    //now we have no need to the array of arr, so we'll delete it.
    delete[] arr;     //deallocate the dynamic array
    return 0;
}
```

Benefits of Dynamic array:
Dynamic array provides memory allocation/deallocation features that’s why we can increase/decrease memory size manually, on the other hand Static memory size is always fixed.

Dynamic array contains information even after completing the function. We can also save memory after completing the tasks by deleting the array.

Question No. 2-a

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

Answer No. 2-a

In general, Class is a user defined data type. It works like an idle template which defines the characteristics of objects, every object holds all the characteristics of a class.

Declaration of an object: syntax- class_name objecct_name;

```
#include <bits/stdc++.h>
using namespace std;
class Qualifications {
    public:
        int position, shot_speed;
        float passing_accuracy, dribbling_accuracy;
};
```

```
int main() {
    Qualifications player;      //declaration a Qualifications type
object named player
    return 0;
}
```

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 one kind of function that helps to initialise the object whose name will be the same as its class. When an object is created then the constructor is called automatically.

Constructor create: syntax - original_class_name(parameters) {}

```
#include <bits/stdc++.h>
using namespace std;
class Qualifications {
    public:
        int position, shot_speed;
        float passing_accuracy, dribbling_accuracy;
         Qualifications(int pos, int spd, float pas, float dri) { //
Constructor
            position = pos;
            shot_speed = spd;
            passing_accuracy = pas;
            dribbling_accuracy = dri;
        }
};
int main() {
    Qualifications player(6, 89, 97.5, 95.5);
    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:
        string name;
        float height;
        int age;
        Person(string nam, float hght, int ag) {
            name = nam;
            height = hght;
            age = ag;
        }
};
int main() {
    Person *p1 = new Person("Alamin", 5.7, 24);
    //cout << p1->name << " " << p1->height << " " << p1->age;
    return 0;
}
```

Question No. 3-a

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

Answer No. 3-a

The size of an object is allocated to the memory that’s equal to the sum of data members size and some other factors.

Question No. 3-b
Can you return a static object from a function? If yes, show with an example.

Answer No. 3-b

Yes, given below the code:

```
#include <bits/stdc++.h>
using namespace std;
class Person {
    public:
        string name;
        float height;
        int age;
        Person(string nam, float hght, int ag) {
            name = nam;
            height = hght;
            age = ag;
        }
};
Person fun() {
    Person per("Alamin", 5.7, 24);
    return per;
}
int main() {
    Person p1 = fun();
    cout << p1.name << " " << p1.height << " " << p1.age;
    return 0;
}
```

Question No. 3-c
Why do we need -> (arrow sign)?
Answer No. 3-c
To access the values of dynamic objects.

Question No. 3-d

Create two objects of the **Person** class from question **2-c** and initialise 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:
        string name;
        float height;
        int age;
        Person(string nam, float hght, int ag) {
            name = nam;
            height = hght;
            age = ag;
        }
};
int main() {
    Person *p1 = new Person("Alamin", 5.7, 24);
    Person *p2 = new Person("Fardin", 5.6, 29);
    if(p1->age > p2->age) cout << p1->name;
    else cout << p2->name;
    return 0;
}
```