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 for storing local variables and function call information.It works as last in first out manner.when a function is called the function and its local variables are pushed into the stack memory, when that function is returned it is removed from the stack as well its all local variables.

Heap Memory : It is a kind of memory where we can perform dynamic memory allocation. During program execution it can dynamically allocate and deallocate the memory space this is why it is more flexible. We can access this memory space using the dereferencing method.

Question No. 1-b

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

Answer No. 1-b

ans : Dynamic memory allocation is important for several reasons:
1. it helps declare variable sized data type during at compile time;
2. We can increase data size according to our need without wasting memory space.
3. It helps efficient memory utilization.
4. It enables data sharing between several functions.
5. We can create objects and data structures during program execution.

```
#include<bits/stdc++.h>
using namespace std;

int main(){
    int *a = new int[5];
    // here we created an dynamic array of size 5
    for(int i=0;i<5;i++){
        cin>>a[i];
    }
    int *b = new int[7];
    // as we need more data to store so we increased the array size and took new array of
size 7
    for(int i=0;i<5;i++){
        b[i] = a[i];
    }
    cin>>b[5]>>b[6];
    for(int i=0;i<7;i++){
        cout<<b[i]<<endl;
    }
    // as here we assigned all the a array elements into b array and increased the size we
can delete the a array and save some space
    delete []a;
    return 0;
}
```

This is how we can create dynamic objects and save space and utilize memory.

Question No. 1-c

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

Answer No. 1-c

ans : we can create a dynamic array in this manner:

```
#include<bits/stdc++.h>
using namespace std;

int main(){
    int *a = new int[5];
    // here we created an dynamic array of size 5
    for(int i=0;i<5;i++){
        cin>>a[i];
    }
}
```

benefits of dynamic array:

1. we can create it in heap memory.
2. its size can be increased easily without losing previous data.
3. when we pass a dynamic array into a function then return it back to the main function its data won't lose.
4. we can delete while compile time by doing so we can save memory.
5. It increases memory utilization.

Question No. 2-a

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

Answer No. 2-a

ans : Class is more like a blueprint depending on this we create an object. Objects are instances of a class. Classes can have multiple data types. class is more like a template that defines attributes and methods that the object of that class can have. Each object under the same class can inherit properties and behaviors of that class from which the object is created. Objects are concrete representations of a class. we can also create dynamic object

```
#include<bits/stdc++.h>
using namespace std;
class Student{
    public:
    int roll;
    char section;
};
```

```
int main(){
    Student s1;
    s1.roll = 1;
    s1.section = 'A';
    cout<<s1.roll<<" "<<s1.section<<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

ans : Constructor is a special kind of function which we need to initialize inside a class as well as it will have the same name as the class name. it is automatically called when an object of that class is created. It is used to initialize object members and other functionalities.Constructor ensure that an object under a certain class is properly initialized. it is responsible for proper memory allocation and resources which are necessary for an object.

constructor creation:

```
#include<bits/stdc++.h>
using namespace std;

class Student{
    public:
    int roll;
    char section;

    Student(int r, char s){
        roll = r;
        section = s;
    }
};
int main(){
    Student s1(1,'A');
    cout<<s1.roll<<" "<<s1.section<<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

```
#include<bits/stdc++.h>
using namespace std;
class Person{
    public:
    char name[1001];
    float height;
    int age;
};


#include<bits/stdc++.h>
using namespace std;
class Person{
    public:
    char name[1001];
    float height;
    int age;

    Person(char *n,float h,int a){
        strcpy(name,n);
        height = h;
        age = a;
    }

};
int main(){
    Person *p1 = new Person("Jonayed",5.75,25);
    cout<<p1->name<<" "<<p1->height<<" "<<p1->age<<endl;
    return 0;
}
```

Question No. 3-a
What is the size that an object allocates to the memory?
Answer No. 3-a
ans : Size of an object depends on various factors. The size of an object depends on the data types it contains. Each data member contributes the overall size of an object. Object size can vary depending on the compiler and platform the user is using to create an object.

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

ans : yes i think i can, it is possible to return a static object from a function in C++. A static object is an object that persists throughout the lifetime of the program and retains its value between function calls.

```
#include<bits/stdc++.h>
using namespace std;

class MyClass {
public:
    int value;

    MyClass(int val) : value(val) {}
};

MyClass& getStaticObject() {
    static MyClass staticObject(42);
    return staticObject;
}

int main() {
    MyClass& obj = getStaticObject();
    cout << obj.value <<endl;

    obj.value = 99;
    cout << obj.value << endl;

    MyClass& obj2 = getStaticObject();
    cout << obj2.value << endl;

    return 0;
}
```

Question No. 3-c

Why do we need -> (arrow sign)?

Answer No. 3-c

ans: The arrow sign (->) is used in C++ to access members of an object through a pointer to that object. It provides a convenient way to access members when working with pointers to objects, especially when dealing with dynamically allocated objects.

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

```
#include<bits/stdc++.h>
using namespace std;
class Person{
    public:
    char name[1001];
    float height;
    int age;

    Person(char *n,float h,int a){
        strcpy(name,n);
        height = h;
        age = a;
    }

};
int main(){
    Person *p1 = new Person("Jonayed",5.75,25);

    Person *p2 = new Person("Mohammad",5.8,24);
    cout<<p1->name<<" "<<p1->height<<" "<<p1->age<<endl;


    cout<<p2->name<<" "<<p2->height<<" "<<p2->age<<endl;

    return 0;
}

#include<bits/stdc++.h>
using namespace std;
class Person{
    public:
    char name[1001];
    float height;
    int age;

    Person(char *n,float h,int a){
        strcpy(name,n);
        height = h;
        age = a;
    }

};
int main(){
    Person *p1 = new Person("Jonayed",5.75,25);

    Person *p2 = new Person("Mohammad",5.8,24);
    if(p1->age>p2->age){
        cout<<p1->name<<endl;
    }
    else{
        cout<<p2->age<<endl;
    }

    return 0;
}
```