# Final Quiz

c/c++205 summer

1. What is the output of the following program?

```
#include <stdio.h>

int main();

void main() {
   printf("OK\n");
}
```

a. OK
b. No output
c. Compile error. We cannot declare main() function.
d. Compile error. Mismatch in declaration & definition.

2. Arrays should always be preferred over Linked lists when:

a. trying to randomly access an element from a list
b. when we do not know the size of the list we are trying to store
c. both cases above
d. when we are only dealing with insertion/deletion

3. When a web server receives requests for pages, sometimes pages have to be generated and on a busy website they cannot all be returned instantly. Therefore, requests that arrive are first stored in memory, then processed by order of arrival. Which data structure seems to you the most appropriate for storing requests before they can be processed:

a. A tree
b. A list
c. A hash table

4. I   m using the following structure to store a list of integer values:

```
typedef struct node {
  int          value;
  struct node *next;
} NODE_T;
```

Which of the following declarations is correct for declaring the head of the list?

a. NODE_T head;

[✓ b. struct node *head = NULL;]

c. struct node head = NULL;

d. void *head = NULL;

e. *NODE_T head = NULL;

5. A macro must always return a value.

a. True

[✓ b. False]

*Actually , macro is not a function , it is just a replacment of string , here the means of question is ,there are two kinds of macro ,*

*1. there is no replacement,just define,such as :*

```
#define X_DEBUG
```

*this can be used while*

```
#ifdef X_DEBUG
...
#else
...
#endif
```

*2. there is a replacement, (it seems like return a value,actually no ),such as:*

```
#define  ADD_X   ((X)+(X))
```

*So, here the answer is : b.False*

*for local object , you can find it in lecture 10 .pag 6 ,the local object (ObjectType y) is defined in func() , it's destructor is automatically called just as option c said*

6. Do local objects need to be deleted?

a. All local objects except pointers need to be explicitly deleted.

b. Yes, any locally declared object has to be deleted.

c. No, the destructor for local objects is automatically called when the curly brace that ends the block is encountered.

d. Only pointers need to be explicitly deleted.

*in lecture 10 .pag5 ,it is mentioned that :" For o2,for which I explicitely called the constructor,the destructor is not called. I should also call it explicitely.*
*here o2 is not the 'local object' , we can see it as a point to a object,it is different.*

7. A copy constructor

a. Should copy all the bytes in the object that is copied

b. Should copy all the bytes that correspond to simple attributes in the object that is copied but set all pointers to NULL

c. Should copy all the bytes that correspond to simple attributes in the object that is copied and recursively copy all allocated memory areas pointed to by the object

*in lecture 10 .pag10~14, the option c is axactly what p14 said :*
*If we want to be safe, we need a copy constructor that performs a "deep copy", which means that it also allocates and copies anything that the original object points to.*