C Dynamic Storage Allocation Answers

Sayak Haldar
IIEST, Shibpur

1) Answer is c) Both a & b

Explanation: Both calloc and malloc function help to obtain block of memory dynamically

2) Answer is d) Both a & b are true

3) Answer is a) zero

Explanation: calloc() returns a storage and every bit of storage is initialized to zero

4) Answer is d) Both b & c

5) Answer is d) welcome

Explanation: It will throw warning for line 4. Since, we do not explicitly typecast the generic pointer returned by calloc.

More Explanation: A character array dynamically created, can be assigned to a string.

But a character array whose memory is allocated statically, cannot be assigned to a string. (It will cause segmentation fault)

Like,

```
char *p=(char*)malloc(100);
p=”welcome”;
printf(“%s\n”,p);
```

It will not cause segmentation fault and successfully print welcome

Whereas,

```
char p[100]=”welcome”;
printf(“%s\n”,p);
```

It will cause segmentation fault.

6) Answer is c) Heap area

7) Answer is b) It is for type-casting

Explanation: Since, malloc returns a void or generic pointer to the initialized storage. But generic pointer could not be referenced. So, we need type-casting

8) Answer is c) free(p)

9) Answer is d) Binary Search Tree

10) Answer is c) The same deallocated memory location

11) Answer is b) free(p1->next) then free(p1)

Otherwise, if the program is huge, then this kind of thing could cause memory leak

More explanation: A memory leak is the gradual loss of available computer memory when a program (an application or part of the operating system) repeatedly fails to return memory that it has obtained for temporary use.

Now, malloc or calloc allocates memory from heap area. Memory allocated from heap area, is not automatically “freed”. So, it could cause potential memory leak if the program is huge. You may face that after running for a while, the executable file generated by your program is unable to allocate more memory from heap area. So, you have to be careful about that.

Now, suppose, you first free (p1) then free(p1->next). Now, after freeing p1, you already lose the pointer p1->next. So, you will not be able to deallocate the memory once allocated for p1->next

12) Answer is d) zero.

Explanation: Since, memory is allocated for p1->next by calloc function, every bit of the initalized storage is automatically assigned to zero. That's why, printf("%d\n", p1->next->x); will print 0.

13) Answer is c) somegrabagevalue

Explanation: Since, the memory allocated for p1->next is done by malloc.

14) Answer is a) true

15) Answer is b) Free the memory pointed to by ptr

References:

1.http://www.sanfoundry.com/c-interview-questions-answers/
2. http://stackoverflow.com/