C Dynamic Storage Allocation Questions

Sayak Haldar
IIEST, Shibpur

1) The function ____ obtains block of memory dynamically.

a) calloc
b) malloc
c) Both a & b
d) free

2)void * malloc(size_t n) returns

a) Pointer to n bytes of initialized storage
b) NULL if the request cannot be satisfied
c) Nothing
d) Both a & b are true

3)calloc() returns a storage that is initialized to

a) Zero
b) Null
c) Nothing
d) One

4) In function free(p), p is a

a) int
b) Pointer returned by malloc()
c) Pointer returned by calloc()
d) Both b & c

5) What is the output of this C code?

```
1.  #include <stdio.h>
2.  int main()
3.  {
4.      char *p = calloc(100, 1);
5.      p = "welcome";
6.      printf("%s\n", p);
7.      return 0;
8.  }
```

a) Segmentation fault
b) Garbage
c) Error
d) welcome

6) Memory allocation using malloc() is done in?

a) Static area
b) Stack area
c) Heap area
d) Both b & c

7) Why do we write (int *) before malloc?
int *ip = (int *)malloc(sizeof(int));

a) It is for the syntax correctness
b) It is for the type-casting
c) It is to inform malloc function about the data-type expected
d) None of the mentioned

8) Which one is used during memory deallocation in C?

a) remove(p);
b) delete(p);
c) free(p);
d) terminate(p);

9) Which of the following will return a result most quickly for searching a given key?

a) Unsorted Array
b) Sorted Array
c) Sorted linked list
d) Binary Search Tree

10. On freeing a dynamic memory, if the pointer value is not modified, then the pointer points to

a) NULL
b) Other dynamically allocated memory
c) The same deallocated memory location
d) It points back to location it was initialized with

11. For the following program, Which of the following should be used for freeing the memory allocated?

```
1. #include <stdio.h>
2. struct p
3. {
4.     struct p *next;
```

```
5.      int x;
6.  };
7.  int main()
8.  {
9.      struct p *p1 = (struct p*)malloc(sizeof(struct p));
10.     p1->x = 1;
11.     p1->next = (struct p*)malloc(sizeof(struct p));
12.     return 0;
13. }
```

a) free(p1);
   free(p1->next)
b) free(p1->next);
   free(p1);
c) free(p1);
d) All of the mentioned

12.What is the output of this C code?

```
1.  #include <stdio.h>
2.  struct p
3.  {
4.      struct p *next;
5.      int x;
6.  };
7.  int main()
8.  {
9.      struct p *p1 = calloc(1, sizeof(struct p));
10.     p1->x = 1;
```

```
11.     p1->next = calloc(1, sizeof(struct p));
12.     printf("%d\n", p1->next->x);
13.     return 0;
14. }
```

a) Compile time error
b) 1
c) Somegarbage value
d) 0

13. What is the output of this C code?

```
1.  #include <stdio.h>
2.  struct p
3.  {
4.      struct p *next;
5.      int x;
6.  };
7.  int main()
8.  {
9.      struct p* p1 = malloc(sizeof(struct p));
10.     p1->x = 1;
11.     p1->next = malloc(sizeof(struct p));
12.     printf("%d\n", p1->next->x);
13.     return 0;
14. }
```

a) Compile time error
b) 1
c) Somegarbage value
d) 0

14. calloc initialises memory with all bits set to zero.

a) true
b) false
c) Depends on the compiler
d) Depends on the standard

15. realloc(ptr, size), where size is zero means

a)Allocate a memory location with zero length
b) Free the memory pointed to by ptr
c) Undefined behaviour
d) Doesn’t do any reallocation of ptr i.e. no operation

References:

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