

A. Questions

1) Declare the following:

- a) A prototype for a function named `func1` that accepts two pointers to `int` as input parameters and does not return anything.
- b) A prototype for a function named `func2` that accepts two pointers to `int` as input parameters and returns a pointer to an `int`.
- c) A prototype for a function named `func3` that accepts a pointer to `int` as input parameter and returns a pointer to an `int`. The function is not allowed to modify the value (pointed to) of the input parameter.
- d) A static double-precision floating point number named `sdouble`.
- e) An `int` variable named `sreg` that has register storage class.

2) Consider the following C snippet:

```
for(int j=0; j<10; j++) {  
    int k;  
    k = j-1;  
}  
int i = j;
```

- a) What is the storage class of `j`?
- b) What is the storage class of `k`?
- c) What is the initial value of `k`?
- d) Is the last statement valid? If so, what is the value assigned to `i`?

3) Consider the following C source file:

```
#include <stdio.h>  
  
void init_x(void)  
{  
    x = 1;  
}  
  
int x;  
  
int main (void)  
{  
    incr_x();  
    printf("%d\n", x);  
    return 0;  
}  
  
void incr_x(void)  
{
```

```
    x++;  
}
```

- What is the storage class of `x`?
- What is the initial value of `x`?
- Can the function `init_x()` access `x` as it is? If not, rewrite `init_x()` so that it can access `x`.
- What is the output of the program?

4) Consider the following C snippet:

```
char *cp;  
cp = (char *)malloc(10*sizeof(char));
```

- Assuming that the allocation is successful, what is the size (in bytes) of the memory block pointed to by `cp`?
- Is it necessary to typecast the return value of `malloc()` to `char *`?
- Rewrite the second line to use `calloc()`.

5) Consider the following C snippet:

```
1  int *ip;  
2  ip = (int *)calloc(5, sizeof(int));  
3  for(int i=0; i<5; i++) {  
4      *ip = i;  
5      ip++;  
6  }  
7  free(ip);
```

Discuss 3 issues with the code.