King Saud University

College of Computer and Information Sciences
Computer Science Department

	Course Code:	**CSC 215**
	Course Title:	**Procedural Programming**
	Semester:	**Fall 2017**
	Final Exam	
	Duration: 120 minutes	
Student Name:		
Student ID:		
Student Section No.		

Computer Science B.Sc. Program: NCAAA: Intended Learning Outcomes (ILO) Student Outcomes ABET: Program Learning Outcomes (PLO) Student outcomes		**Question No. Relevant is Hyperlinked**	**Covering %**
NCAAA	**1. Knowledge (NCAAA)** Suggested verbs (list, name, record, define, label, outline, state, describe, recall, memorize, reproduce, recognize, record, tell, write)		
ABET	(i) Use current techniques, skills, and tools necessary for computing practices; *The students learn how to use Integrated Development Environment to compile and run C programs. Students also learn the differences between procedural and object oriented languages*		
NCAAA	**2. Cognitive Skills (NCAAA)** Suggested verbs (estimate, explain, summarize, write, compare, contrast, diagram, subdivide, differentiate, criticize, calculate, analyze, compose, develop, create, prepare, reconstruct, reorganize, summarize, explain, predict, justify, rate, evaluate, plan, design, measure, judge, justify, interpret, appraise)		
ABET	b. An ability to analyze a problem, and identify and define the computing requirements appropriate to its solution; *Students learn how to read and analyze a problem, determine its input and output, and allocated the required storage statically or dynamically.*	Q2 , Q4	
	c. An ability to design, implement and evaluate a computer-based system, process, component or program to meet desired goals. *Students learn how to analyze a procedural c program, evaluate c expressions and interpret algorithmic steps from natural language into c programs.*	Q1	

Question1: Write True/ False **(Total 10 points)**

	Statement	True or False
1	In the absence of explicit initialization: automatic and static variables are set to zero.	**False**
2	When an integer is added to a pointer, the pointer is incremented by that integer times the size of the object to which the pointer refers.	**True**
3	If `x` and `y` are defined as two integers with initial value for each; and if it is given that `!x == !y`, then it is safe to infer that `x == y`	**False**
4	The statement `scanf("%d", x);` will cause a compiler warning or error, no matter what the type of `x` is.	**False**
5	The `for`-loop counter must be declared as `int`. It cannot be declared as `float`.	**False**
6	To wipe out the contents of file `"f.txt"` it is enough to execute the statement: `fclose(fopen("f.txt","w"));`	**True**
7	The size of the array: `short arr[4][3]={{1},{2,3},{4,5,6}};` in memory is 12 bytes.	**False**
8	If `int i;` the following statement prints nothing to the output: `if (i=1, i > 0, i--) printf("Hello");`	**False**
9	The standard output stream is normally connected to the screen but can be redirected to a file.	**True**
10	If `f1` and `f2` are two float variables; comparison of `f1` and `f2` using == or `!=` should be avoided.	**True**

Question 2: Select the correct answer **(Total 10 points)**

1	2	3	4	5	6	7	8	9	10
C	**A**	**B**	**A**	**D**	**D**	**C**	**B**	**B**	**D**

1. What is the output of the following program (if any)?

```
#include "stdio.h"
#include "string.h"
int main(){
 printf("%d %d",sizeof("string"),strlen("string"));
 return 0;
}
```

A. `6 6` B. `7 7` C. `7 6` D. None of the above

2. To convert a `float f = 1.0;` to the integer value (1), one can use:

A. `int i1 = (int) f;`
B. `int i2 = * (int *) &f;`
C. `A or B`
D. None of the above

---

3. What is the output of the following program (if any)?

```
#include "stdio.h"
int main(){
  static int none;
  int y = call(none);
  printf("%d ",y);
  return 0;
}
int call(int z){
   return ++z;
}
```

A. `0` B. `1` C. Undefined value D. None of the above

---

4. What is the output of the following program (if any)?

```
#include "stdio.h"
int main(){
   int i=0;
   if(!i){
      i=((5,(i=3)),i=1);
      printf("%d",i);
   }
   else printf("equal");
  return 0;
}
```

A. `1` B. `3` C. `5` D. `equal`

---

5. If storage of type `int` takes 32 bits, what is the output of the following program (if any)?

```
#include <stdio.h>
struct abcd {
  int a;
  char b;
  int c;
  char d;
  short e;
} st;
int main() {
  printf("%d",sizeof(st)); return 0;
}
```

A. `11` B. `12` C. `14` D. `16`

---

6. What is the output of the following program (if any)?

```
#include<stdio.h>
#include<stdlib.h>
void populate(int*a){
  int*parr = (int*)malloc(2*sizeof(int));
  parr[0] = 37;
  parr[1] = 73;
  a = parr;
}
int main(){
  int *a = NULL;
  populate(a);
  printf("a[0] = %d and a[1] = %d\n", a[0], a[1]);
  return 0;
}
```

A. 37 and 73 B. 73 and 37 C. 0 and 0 D. None of the above

---

7. What is the output of the following program (if any)?

```
#include <stdio.h>
int main(){
  char str[] = "GHIJKLMN";
  char* ptr = str;
  *ptr = *ptr + 2;
  ptr = ptr + 2;
  printf("%c", *ptr);
  ptr--;
  printf("%c", *ptr);
  ptr = str;
  printf("%c", *ptr);
  return 0;
}
```

A. KJG B. JIG C. IHI D. None of the above

---

8. How many times the word "Hi" is printed?

```
#include <stdio.h>
int main(){
  int i = 0, j = 0;
  for (i = 0; i < 5; i++){
    for (j = 0; j < 4; j++)
      if (i > 1)
        break;
      printf("Hi\n");
  }
  return 0;
}
```

A. 4 times B. 5 times C. 7 times D. 9 times

---

9. What is the output of the following program (if any)?

```
#include <stdio.h>
int main(void){
  const int N = 3;
  int i, j;
  for (i = 1; i <= N + 1; i++){
    for (j = 1; j <= N; j++)
      printf("%c", 'A' + ((i + j - 2) % N));
    printf("\n");
  }
  return 0;
}
```

A.
```
ABC
BCA
CAB
```

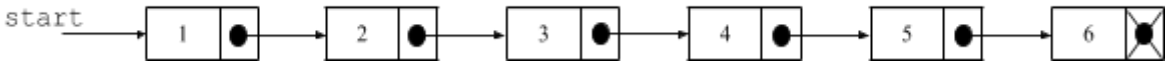
B.
```
ABC
BCA
CAB
ABC
```

C.
```
ABC
BCD
CDE
```

D.
```
ABC
BCD
CDE
DEF
```

---

10. If `start` points to the first node in the following linked list:



What is the output when `fun(start);` is called?

```
:
void fun(struct node* p) {
  if (!p)
    return;
  printf("%d ", p->data);
  if (p->next)
    fun(p->next->next);
   printf("%d ", p->data);
}
```

A. `1 2 3 4 5 6` B. `1 3 5` C. `1 3 5 5 3` D. `1 3 5 5 3 1`

Question 3: Answer the following questions based on its given code:**(Total 14 points)**

A. Assume a is stored starting in memory at address `0x8049000`. (4 points)
What is the value stored in p for the following cases:

	Case	Value of p
1	`int a[10];` `int* p = &a[7];`	**0x804901C**

2	`struct foo{`
	`  int x;`
	`  int y;`
	`  int z; } a;`
	`int* p = &a.y;`
		0x8049004

3	`struct foo{`
	`  int x[4];`
	`  int y[4]; } a;`
	`int* p = &a.y[2];`
		0x8049018

4	`struct foo{`
	`  int x[4];`
	`  int y[4]; } a[10];`
	`int* p = &a[7].y[2];`
		0x80490F8

B. Complete the following function by filling in blanks. (4 points)
pack takes a linked list as an argument (whose nodes are potentially allocated at arbitrary points in heap memory), and returns an identical linked list where all list nodes are adjacent in heap memory.

```
01 struct list { int data; struct list *next; };
02 struct list* pack(struct list* ll) {
03   int size, i;
04   struct list* all;
05   for (size=0, all=ll; all; size++, all=all->next);
06    /* Allocate storage for the resulting list */

07   all = (struct list*)malloc(size*sizeof(struct list));
08   for (i = 0; i < size; i++, ll = ll->next){
09     /* copy data from the original */

10     all[i].data = ll->data;
11     /* point next to the next adjacent location */

12     all[i].next = all+i+1;
13   }
14   /* terminate the pointer chain */

15   all[i-1].next = NULL;
16   return all;
17 }
```

C. Complete the following program by filling in blanks. (6 points)

Function `modify` receives an integer array `arr` of size `m`. It swaps each element in `arr` with another element from `arr` at a random index generated by calling `rand()`. It returns `0` if the array is not modified otherwise it returns `1`.

```
01 #include<stdio.h>
02 #include<stdlib.h>

03 int modify(int* arr, int m);
04 int main() {
05   int x[8]={1,2,3,4,5,6,7,8};

06   if( modify(x,8) )
07     printf("array modified");
08   else printf("array not modified");
09   return 0;
10 } //below is the function definition of modify

11 int modify(int* arr, int m){
12   int rand_index, temp, i;
13   /* array arr will not be modified if its size is 1 */

14   if (m == 1) return 0;
15   for(i = 0; i< m; i++){
16     /*Swap element at index i with element at random index */

17     rand_index = rand()%m;
18     temp = arr[rand_index];

19     arr[rand_index] = arr[i];
20     arr[i] = temp;
21   }
22   return 1;
23 }
```

Question 4: The following functions: **(Total 6 points)**

```
float max(float* a, int n);
float min(float* a, int n);
float avg(float* a, int n);
```

return maximum value, minimum value and average value of array `a` of size `n` respectively.

A. Implement the function `apply_stat` that takes an array of float, its size, and one of these three functions as parameters, and returns the value of the corresponding statistic.

```
float apply_stat(float* a, int n, float (*f)(float*, int)){
  return f(a,n);
}
```

B. Write one statement that prints the average value of array `fa` using function `apply_stat`.

```
float fa[] = {23.56, 31.82, 32.90, 18.24, 65.75, 70.70};
printf("%.2f\n", apply_stat(fa, 6, max));
```

Bonus Question (2 points)

The stack structure is utilized through the functions:

```
int is_empty(struct stack s);

int peak(struct stack s);

void push(struct stack* s, int data);

int pop(struct stack* s);
```

If stack S contains non-negative integer values write a function `pop_min` that removes the minimum value from the stack S and returns it. All other values of the stack must be, after the function `pop_min` returns, in their original relative order.

```
int pop_min(struct stack* s){
  int min, val;
  struct stack* s2 = (struct stack*)calloc(1, sizeof(struct stack));
  if (!is_empty(s))
    min = pop(s);
  else
    min = -1;
  while (!is_empty(s)){
    if ((val = pop(s)) < min)
      min = val;
    else
      push(s2, val);
  }
  while (!is_empty(s2))
    push(S, pop(s2));
  return min;
}
```