Section A

[5 marks]

1. There are ten stacks of coins. Each stack contains ten coins. All of the coins in one stack are counterfeit. All the coins in other stacks are genuine. Every genuine coin weighs 7 grams. Every counterfeit coin weighs 8 grams. You have a digital weighing scale that can accurately measure and display the weight in grams of any number of coins. We want to identify the stack of counterfeit coins. However, we also have the restriction that we can use the weighing scale only once. Explain how you will identify the stack of counterfeit coins.

Answer:

Number the coin stacks from 1 to 10. Take 'x' number of coins from stack number 'x'. We will pick up (1+2+3……+10)= 55 coins. If all the stacks had genuine coins then total weight of these 55 coins would be 55X7 = 385 grams.

Let us assume that total weight of 55 coins comes out to be 'z' grams. Let us also assume that stack number 'y' has counterfeit coins. Then total weight of 55 coins will be 385+y grams. Thus we have

z = 385 + y

We can identify the stack of counterfeit coins in one weighing.

2. For the following code snippet, choose the equivalent code snippet:

```
for(i=5; i<n; i++){  x=i+1;   }
```

```
A. i=5; while(i<n){ i++; x=i+1;}
B. i++; while(i<n){ i=5; x=i+1;}
C. while(i<n){ i=5; i++; x=i+1;}
D. i=5; while(i<n){ x=i+1; i++;}
```

Correct answer D

3. Consider following code snippet. What will be the value of z?

```
int x=57, y=9, z; z=x/y;
```

Answer: 6

4. Is 'main' a keyword in C?

Answer: No

5. In case of for loop, which of the following will result in an infinite loop?

A. absence of looping condition

B. absence of initialization expression

C. presence of multiple looping conditions

D. C compiler does not allow an infinite loop as available memory is finite.

Answer: A

6. If ptr is pointer variable, then *ptr is called dereferencing the pointer. **ptr is called two levels of dereferencing. While using pointers in general, dereferencing operation can be applied upto how many levels?

a. any number of levels depending on support from the compiler

b. only one

c. size of the data type

d. 32 times on 32 bit system and 64 times on a 64 bit system

Answer: A

Section B

Q 1. In the following code, what is the size of the array arr?

```
char arr[] = "hello, world!";
```

14

Q 2. What is the output of the following piece of code?

```
char arr[] = "hello, world!";
arr[5] = ’\0’;
printf("%s", arr);
```

hello

Q 3. What is the output of the following piece of code?

```
char arr[] = "hello, world!";
printf("%10s", arr);
```

hello, world!

Q 4. What is the output of the following piece of code?

```
char arr[] = "hello, world!";
printf("%.5s", arr);
```

hello

Q 5. What is the output of the following piece of code?

```
char arr[] = "hello, world!";
char *p;
p = arr;
while(*++p);
printf("%ld", p - arr);
```

13

Q 6. What is the output of the following piece of code?

```
char arr[] = "hello, world!";
char *p;
p = arr;
while(*++p);
printf("%s", p);
```

Nothing will be printed

Q 7. What is the data type of `&arr` in the following piece of code?

```
int *arr;
```

`int **` or pointer to pointer to `int`

Q 8. What is the data type of `&arr[0]` in the following piece of code?

```
int *arr[10];
```

`int **` or pointer to pointer to `int`

Q 9. Borrowing the notation from the lectures, what is the content of the queue after the following operations have been performed?

```
abcPPPPdefPP
```

f

Q 10. Borrowing the notation from the lectures, how many nodes does a linked list contain if `start.pointer.pointer` is equal to `NULL`?

2

Q 11. What is the output if the following piece of code is executed as `./a.out 1 2 3`?

```
#include <stdio.h>

int main(int argc, char *argv[])
{
  printf("%d", argc);
  return 0;
}
```

4

Q 12. What is the output if the following piece of code is executed as `./a.out 1 2 3`?

```
#include <stdio.h>

int main(int argc, char *argv[])
{
  printf("%s", argv[2]);
  return 0;
}
```

2

Q 13. If the file `abc.txt` does not exist, what does the following `fopen` return?

```
fopen("abc.txt", "r");
```

NULL

Q 14. If the file abc.txt does not exist, what does the following fopen return?

```
fopen("abc.txt", "r+");
```

NULL

Q 15. What is the output of the following piece of code if the file abc.txt exists?

```
FILE *fp;
fp = fopen("abc.txt", "r+");
printf("%ld", ftell(fp));
```

0

Q 16. What is the constant that has to be provided to fseek to seek from the beginning of a file?

SEEK_SET

Q 17. Will the following piece give compilation error or print the address of s?

```
struct str {
  struct str *p;
};
struct str s;
s.p = &s;
printf("%p", s.p);
```

Address of s

Q 18. Will the following piece give compilation error or print the address of s?

```
struct str {
  struct str *p;
};
struct str s;
s.p = &s;
printf("%p", s.p->p);
```

Address of s

Q 19. What is the output of the following piece of code on input 50t?

```
int a, b;
char c;
scanf("%d%d%c", &a, &b, &c);
printf("%d %d %d %c", a, b, c);
```

50 Garbage value Garbage value Garbage value

Q 20. What is the output of the following piece of code on input 50t?

```
int a, b, i;
char c;
i = scanf("%d%d%c", &a, &b, &c);
printf("%d", i);
```

1

Q 21. What is the output of the following piece of code on input 50t?

```
int a, b;
char c;
scanf("%d", &a);
scanf("%d", &b);
scanf("%c", &c);
printf("%d %d %d %c", a, b, c);
```

50 Garbage Value ASCII value of t(116) Garbage Value

Q 22. What is the output of the following piece of code?

```
printf("%d", printf("%s", "%d"));
```

%d2

Q 23. What will be the state of the array after the first round of radix sort?

21 90 42 100 25 42 41

90 100 21 41 42 42 25

Q 24. State whether the following postfix expression is correct. If yes, evaluate and state the answer. All the numbers given consists of single digits.

```
1 2 3 4 * + 5 5 * -
```

Incorrect postfix expression

Q 25. State whether the following postfix expression is correct. If yes, evaluate and state the answer. All the numbers given consists of single digits.

```
1 2 3 4 * * + 5 5 * -
```

0

CS101 Introduction to Computing (May 2018)

Answer sheet Sections C and D

Roll No:	
Name	

Section C: Questions 1 to 14

Ques. No	Part A	PART B: Reason, Explanation or Output
1	E	10
2	E	2
3	C	No output will be emitted
4	D	There is no memory for int
5	B	10
6	E	20
7	E	Hello from Prime Mr Minister Ram Kumar
8	B	Warnings: Uninitialised day, month, year. Or, No allocated memories. Runtime failure.
9	E	11 12 YR17
10	E	Comp(4) = 8
11	E	Input = 1357; output = 16
12	D	Unpredictable. Not enough space for any strings! Definite wrong if date.
13	E	No output will appear! (Attempt to write on a read-only open file does not terminate program Only the write does not occur.)
14	1 2 3 5 * + + 1 2 3 * 2 3 + * + +	

Student must test their code(s) on a computer before making any claim of a wrong or incorrect assessment for a section C or D questions.

In many books Q13 may have been marked as correct for answer that is not appropriate. This mistake will be corrected if noted during a remarking request by the student (that is, the over-awarded score will be reduced by 1 to correct the mistake, if noticed).

Section D: Questions 15 to 25

Q.	Code	
15	a b c * d + + Following answer is also accepted: ERROR unexpected symbol.	No marks If stack or its index is accessed directly. A valid access to stack is only through pop() and push(). No marks if the input symbol is not checked to make sure that it was a "welcomed" input. Symbols taken out of stack by pop() but not needed/consumed must be pushed back into the stack. Students must use a computer and run their code (as in their answer) before asking for any remarking.
16	a b ERROR unexpected operator. Also accepting answer: ERROR unexpected symbol.	
17	if (!hasWelcome(c)) return -1;	

18	if (!hasWelcome(c)) return -1;
	while ((c = pop()) != '(')
	        printf(" %c ", c);	

19	if (!hasWelcome(c)) return -1;
	if ((stk = pop()) == '*')
	        printf(" *");
	else if (stk != EOF)
	     push(stk);	

20	if (!hasWelcome(c)) return -1;
	if ((stk = pop()) == '*')
	        printf(" *");
	else if (stk != EOF) push(stk);
	if ((stk = pop()) == '+')
	        printf(" +");
	else if (stk != EOF)
	        push(stk);	

21	if (!hasWelcome(c)) return -1;
	while ((c = pop()) != EOF)
	    if (c == '(')
	        {
	           printf(" ERROR!\n");
	           return -1;
	        }
	    else
	        printf(" %c", c);	

22	readNumber = scanf("%[0-9]", remainingNumber);	
23	if (readNumber) printf("%s", remainingNumber);	
24	if (c<0) return 1;	
25	if (c==0) return 0;	