Deepankar Sharma
233512013

Course code: 230MC103
Course title: Programming and Problem Solving

Assignment 1

Part A

Ques 1. (a) Garbage value
Ques 2. (b) Address value
Ques 3. (c) 20
Ques 4. (c) 0
Ques 5. (a) 10
Ques 6. (b) Top Down
Ques 7. (c) Compile Error
Ques 8. (b) ouworld
Ques 9. (d) all of the above
Ques 10. (b) 25 26 27

Part B

Ques (1) Preprocessor directives in C

Preprocessor directives in C are the special instructions that are processed before the compilation of the actual C code. They are used to include or exclude certain parts of the code, define constants. Preprocessor directives start with '#'. Some commonly used preprocessor directives are:

(1) #include <stdio.h>;
It is used to include header files
(2) #define x 4;
It is used to define the macros. For example value of x will be 4 (constant)
(3) #error : used to generate error message
(4) # warning : used to generate warning message
(5) #ifdef , #ifndef , #endif , #else : used for conditional compilation
(6) #pragma : provides non-standard compiler specific instructions
(7) # pragma once : ensures header file is included only once.

Deepankar Sharma
233512013
Sharma.

Ques 2

Loops in C

① **for loop** : for loop is used when you know how many times you need to iterate.
② **while loop**: while loop is used when you want iteration to continue until a given condition is true.
③ **Do-while loop** : Do-while loop is similar to while loop but you want the body of loop to execute once even if the condition isn't true.

```
for (int i=0; i<5; i++){
    printf("%d", i);
}
```

```
int i=0;
while (i<5){
    printf("%d", i);
    i++;
}
```

```
int i=0;
do {
    printf("%d ", i);
    i++;
} while(i<5);
```

Ques 3

Nested If Else

Nested If-Else is also called as the Nested-If Else ladder. Basically you have another if else statement inside a given if-else statement.

```
#include <stdio.h>
int main() {
    int x=10; int y=5;
    if (x>5) {
        printf("x is greater than 5");
        if (x>10)
            printf("x is greater than 10");
        else printf("x is not greater than 10");
    }
    else
        printf("x is not greater than y");
    return 0;
}
```

Ques 4 call-by-value vs call-by-reference

In call by value, a local copy of the variable is created within the function block which doesn't affect its value in the parent block from where function is called.

In call by reference, a pointer is passed to function which points/references to same variable in the parent block. Every operation is applied to original variable itself.

```
#include<stdio.h>
void cbyvalue(int x){
        x*=10;
}
void cbyreference(int *x){
        *x*=10;
}
int main(){
    int x=5;
    printf("%d ",x);
    cbyvalue(x); //5
    printf("%d",x);
    //cbyref
    cbyreference(&x); //50
    printf("%d",x);
    return 0;
}
```

Deepankar Sharma
2335 12013
Forma.

Deepankar Sharma

Ques 5

user-defined functions

User defined functions in C are very important as they are used to effectively modularize the code instead of repeating the same lines of code all over again. Also user defined functions promote abstraction and code reusability.

```
#include <stdio.h>
int areax(int a, int b){
        return a*b;
}
int main(){
    int length=10;
    int breadth=20;
    int area;
    area=areax(length,breadth); return 0;
}
```

Ques 6

Find largest number in an array

```
#include<stdio.h>
int main(){
    int max=-1, pos=-1;
    int arr[]={1,5,7,8,10};
    for(int i=0; i<5; i++){
        if(arr[i]>max){
            max=arr[i];
            pos=i;
        }
    printf("The maximum element in arr is %d at index %d",
            arr[pos], pos);

    return 0;
}
```

Deepankar Sharma
233512013

Ques 7

vowels in string

```
#include <stdio.h>
int main() {
    char str[] = "This is great";
    int count=0; int pos=0;
    for( int i=0; i<20; i++) {
        switch (str[i]){
            case 'a' : pos+=1 ; break;
            case 'e' : pos+=1 ; break;
            case 'i' : pos+=1 ; break;
            case 'o' : pos+=1 ; break;
            case 'u' : pos+=1 ; break;
        }
    }
    printf("# vowels is %d ", pos);
    return 0;
}
```

Ques 8

(a) **Global variables & local variables**

Global variables and local variables differ in term of their scope, lifetime and accessibility. Global variables are accessible throughout the program whereas the local variables are confined to the local function block or the code block.

Actual & formal parameters

Formal parameters are the signatures or placeholder defined in the function's parameter list. They are used within the function definition. Actual parameters are the actual variables that are passed as arguments to the function call. The function actually operate on the values of actual parameters.