Model Answer – Odd Semester Exam -2015
Subject: Programming in 'C' Language (MSC-103)
Class: M.Sc.-I (Computer Science)

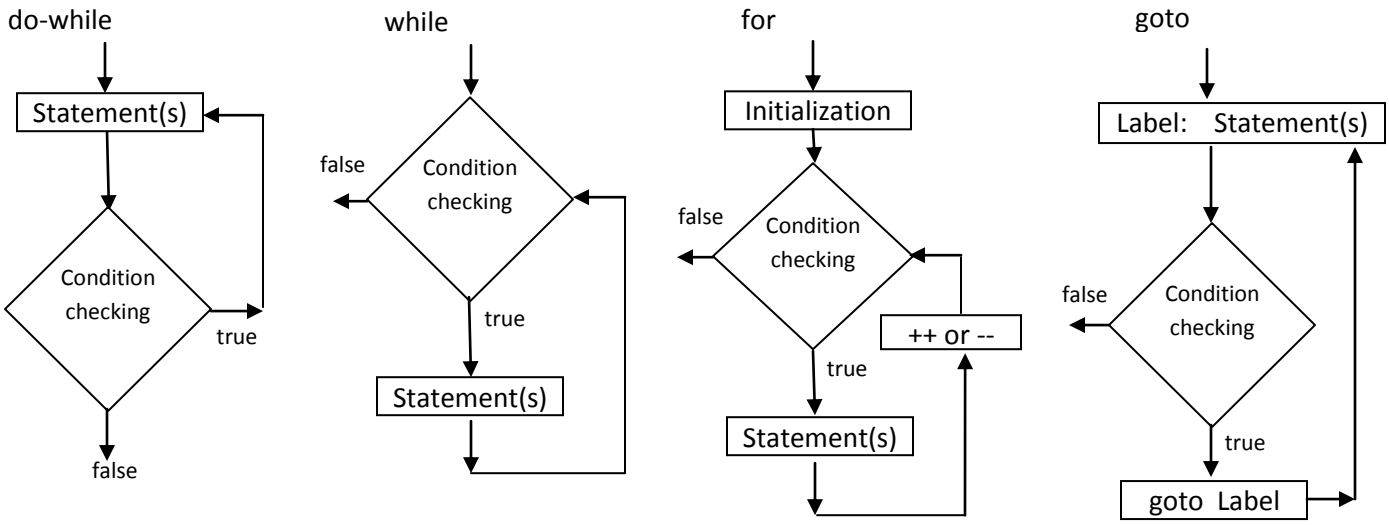
1.A. i) a ii) a iii) d iv) c
v) a vi) d vii) c

1B. int - %d
float - %f
char - %c

1C. Inbuilt union REGS is used to pass information to the functions int86, int86x, intdos, intdosx. It can also used to fetch information from the above functions.
The inbuilt union REGS structure:

```
union REGS
{
        struct WORDREGS x;
        struct BYTEREGS h;
}
struct WORDREGS
{
        unsigned int ax,bx,cx,dx;
        unsigned int si,di,cflag,flag;
}
struct BYTEREGS
{
        unsigned char al,ah,bl,bh;
        unsigned char cl,ch,dl,dh;
}
```

2i. Keywords used to construct loops in c:
do,while,for
Other than these we can create loops by using goto keyword with appropriate use of label of statements and conditional statement.
Flow control:



2ii.

```
#include<conio.h>
#include<stdio.h>
void main()
{
        int c=0,n=1;
        clrscr();
        label: printf("%d ",n+=2);
        c++;
        if(c<10)
                goto label;
        getch();
}
```

3i.

Strcpy(): Copies all character of source string to the destination string including the terminating character('\0') It returns destination string. It takes two arguments: destination string as character pointer, destination string as constant character pointer.
Strncpy: Copies first n characters of source string to the first n destination string character and does not add the terminating character. It returns destination string. It takes three arguments: destination string as character pointer, destination string as constant character pointer, n integer to specify number of characters.
Strlwr: Converts a string to lower case string. Work only on the characters a to z. Returns the pointer to the converted string. It has only one argument as character string.
Strupr: Converts string to upper case string. Work only for the characters a to z. Returns the pointer to the converted string. It has only one argument as character string.
Strcat: It appends source string to the end of destination string. The length of destination string is sum of length of source and destination string. It returns the pointer of the destination string. It takes the destination string, source string as arguments.
Source code for string functions:

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
void main()
{
        char *src="HELLO WORLD",dest[15];
        clrscr();

        strcpy(dest,src);
        printf("%s\n",dest);// HELLO WORLD

        strncpy(dest,src,5);
        dest[5]='\0';
        printf("%s\n",dest);// HELLO

        printf("%s\n",strupr("text to upper"));// TEXT TO UPPER

        printf("%s\n",strlwr(dest));// hello

        strcat(dest,src);
        printf("%s\n",dest);// helloHELLO WORLD

        getch();
```

```
}
```
Output:
HELLO WORLD
HELLO
TEXT TO UPPER
hello
helloHELLO WORLD

3ii.

```
#include<conio.h>
#include<stdio.h>
#include<string.h>
void main()
{
	char *src,*rev;
	clrscr();

	printf("Give a string: ");
	gets(src);

	strcpy(rev,src);
	if(strcmp(src,strrev(rev))==0)
		printf("<%s> is palindrome as the reverse is <%s>",src,rev);
	else
		printf("<%s> in not palindrome as the reverse is <%s>",src,rev);
	getch();
}
```
Output 1:
Give a string: MALAYALAM
<MALAYALAM> is palindrome as the reverse is <MALAYALAM>

Output 2:
Give a string: hello
<hello> in not palindrome as the reverse is <olleh>

4i.

```
#include<stdio.h>
#include<conio.h>

#define PI 3.14285
#define AREACIRCLE(x) PI*x*x
void main()
{
	float r=7.0,a=0.0;
	clrscr();

	printf("Area=%f", AREACIRCLE(r));
	getch();
}
```

4ii.

Some of the Preprocessor directives are used to achieve conditional compilation. Before the source code enters into the compiler, it passes through the cpp(C pre processor) application. In cpp, filtration of coding is done as per the conditional compilation directives used in the source code. Apart from that, the macro expansion is also done in cpp.

Source code explaining preprocessor directives used for conditional compilation:

```
#include<stdio.h>
#include<conio.h>

#define INTEL 0
#define MOTOROLA 0
#define IBM 1
#define BLOCK

void main()
{
        #if INTEL
                int x=100;
                clrscr();
                printf("intel code....%d\n",x);
        #elif MOTOROLA
                int x=200;
                clrscr();
                        printf("motorola....%d\n",x);
        #elif IBM
                int x=300;
                clrscr();
                printf("IBM...%d\n",x);
        #endif

        #ifdef BLOCK
                printf("BLOCK is defined ...\n");
        #endif

        getch();
}
```

Output:

```
IBM…300
BLOCK is defined
```

5i.

as per the class note given.

5ii.

```
#include<conio.h>
#include<stdio.h>
void main()
{
        FILE *fp;
        int ch;
        int f='X', r='Y'; //find character 'X' and replace with 'Y'

        clrscr();
```

```
		fp=fopen("d:\\test.txt","r+");
		if(fp==NULL)
		{
			perror("Source File Access Error due to ... ");
			getch();
			exit(0);
		}

		while((ch=fgetc(fp))!=EOF)
		{
			if(ch==f)
			{
				fseek(fp,-1,SEEK_CUR);
				//fflush(stdin);
				fputc(r,fp);
				fseek(fp,0,SEEK_CUR);
			}
		}
		printf("file modified... %c to %c",f,r);

		fclose(fp);
		getch();
	}
```
Output: (if program completed successfully)

File modified… X to Y

6i.

```
 #include <graphics.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#include <dir.h>
#include<dos.h>

union REGS in,out;

int initMouse(void);
void showMousePointer(void);

int main(void)
{
	/* request auto detection */
	int gdriver = DETECT, gmode, errorcode;
	clrscr();

	/* initialize graphics mode */
	initgraph(&gdriver, &gmode,"c:\\turboc3\\bgi");

	/* read result of initialization */
	errorcode = graphresult();
	if (errorcode != grOk)  /* an error occurred */
```

```
	{
		printf("Graphics error: %s\n", grapherrormsg(errorcode));
		printf("Press any key to halt:");
		getch();
		exit(1);/* return with error code */
	}

	if(!initMouse())
	{
		outtextxy(1,1,"\nMouse failed to initialized....");
		getch();
		closegraph();
		exit(0);
	}

	showMousePointer();

	getch();
	closegraph();
	return 0;
}

int initMouse()
{
	in.x.ax=0;  ////check mouse status
	int86(0x33,&in,&out); // invoke interrupt
	return out.x.ax;
}
void showMousePointer()
{
	in.x.ax=1;//show mouse pointer
	int86(0x33,&in,&out);
}
```

Output: mouse pointer is shown in graphics mode.

6ii.

```
	#include<stdio.h>
	#include<conio.h>
	void main()
	{
		struct student
		{
			char name[25];
			int age;
		};
		int i;
		struct student s[3];
		clrscr();
		for(i=0;i<3;i++)
		{
			printf("Enter name: ");
			gets(s[i].name);
```

```
                printf("Enter age: ");
                scanf("%d",&s[i].age);
                fflush(stdin);
        }
        printf("Name\t\tAge\n------------------------------\n");
        for(i=0;i<3;i++)
                printf("%s\t\t%d\n",s[i].name,s[i].age);
        getch();
}
```

OUTPUT:

```
Enter name: pomy
Enter age: 22
Enter name: joy
Enter age: 25
Enter name: dasu
Enter age: 24
Name            Age
---------------------------------
pomy            22
joy             25
dasu            24
```

7i.

```
#include<stdio.h>
#include<conio.h>

int plus(int,int);
int minus(int,int);
int cross(int,int);
int wrapperOP(int (*funPtr)(int,int),int x,int y);

void main()
{
        clrscr();

        //call to different functions a using a single wrapper function
        printf(" add = %d\n",wrapperOP(plus,10,20));
        printf(" sub = %d\n",wrapperOP(minus,10,20));
        printf(" cross = %d\n\n",wrapperOP(cross,10,20));

        getch();
}
int plus(int x,int y)
{
        return x+y;
}
int minus(int x,int y)
{
        return x-y;
}
int cross(int x,int y)
{
```

```
	return x*y;
}
int wrapperOP(int (*funPtr)(int,int),int x,int y)
{
	return funPtr(x,y);
}
```

7ii.

malloc: Allocates memory for data from program memory heap. The only argument is the size of memory.
calloc: Allocates memory for data from main i.e. C memory heap and initializes the data. It has two arguments to pass: one is number of space and second is the size.
realloc: Adjusts the size of already allocated memory through either malloc or calloc and relocate them . It copies the old data values into new. It does not initialize new data locations.

```
#include<conio.h>
#include<stdio.h>
void main()
{
	long *arr,co=100,*c,*ra;
	int i,j,n;
	clrscr();

	printf("How many nos : ");
	scanf("%d",&n);

	arr=(long*)malloc(n*sizeof(long));
	for(i=0;i<n-1;i++,co+=10)
		*(arr+i)=co;

	printf("\nmalloc \n");
	for(i=0;i<n;i++)
			printf("%ld-%u  ",arr[i],&arr[i]);

	c=(long*)calloc(n,sizeof(long));//calloc does the same thing as malloc and initializes the data
	printf("\ncalloc \n");
	for(i=0;i<n;i++)
			printf("%ld-%u  ",c[i],&c[i]);

	ra=(long*)realloc(arr,(n+2)*sizeof(long));
	printf("\nrealloc \n");
	for(i=0;i<n+2;i++)
			printf("%ld-%u  ",ra[i],&ra[i]);

	printf("\nrealloc removes the previous ptr after copy \n");
	for(i=0;i<n;i++)
			printf("%ld-%u  ",arr[i],&arr[i]);

	getch();
}
```

8i.

drawpoly: Draws the outline of a polygon as per current line, style and colour then fills the polygon as per current fill pattern and fill colour. It takes two arguments: number of points, coordinate of points as integer array of size twice number of points. It returns none.

ellipse: Draws the outline of an elliptical arc as per current line colour then fills the ellipse as per current fill pattern and fill colour. It takes six arguments: center-x, centre-y, start angle, end angle, x-radius,y-radius. It returns none.

sector: Draws the elliptical pie slice of an elliptical arc as per current line colour then fills the elliptical pie slice as per current fill pattern and fill colour. It takes six arguments: center-x, centre-y, start angle, end angle, x-radius,y-radius. It returns none.

8ii.

```
#include <stdio.h>
#include <conio.h>
void fun(int);
        void main(void)
{
        int i;
        clrscr();

        fun(100);
        getch();
}
void fun(int n)
{
        if(n>1) fun(n/2);
        printf("%d",n%2);
}
```

Output: 1100100